On the Polling Problem for Social Networks

Hoang Bao Thien, Abdessamad Imine

On the Polling Problem for Social Networks

Hoang Bao Thien and Abdessamad Imine

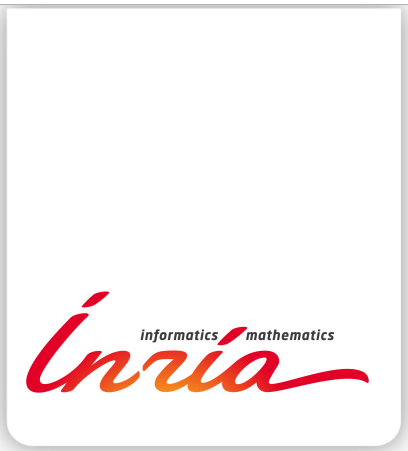
RESEARCH REPORT

N° 8055

September 2012

Project-Team CASSIS

ISSN 0249-6399 ISRN INRIA/RR--8055--FR+ENG



On the Polling Problem for Social Networks

Hoang Bao Thien* and Abdessamad Imine †

Project-Team CASSIS

Research Report n° 8055 — September 2012 — 28 pages

Abstract: We tackle the polling problem in social networks where the privacy of exchanged information and user reputation are very critical. Indeed, users want to preserve the confidentiality of their votes and to hide, if any, their misbehaviors. Recent works [9, 10] proposed polling protocols based on simple secret sharing scheme and without requiring any central authority or cryptography system. But these protocols can be deployed safely provided that the social graph structure should be transformed into a ring-based structure and the number of participating users is perfect square. Accordingly, devising polling protocols regardless these constraints remains a challenging issue.
In this work, we propose a simple decentralized polling protocol that relies on the current state of social graphs. More explicitly, we define one family of social graphs and show their structures constitute necessary and sufficient condition to ensure vote privacy and limit the impact of dishonest users on the accuracy of the output of the poll. In a system of N users with $D \leq N/5$ dishonest ones (and similarly to the works [9, 10] where they considered $D < \sqrt{N}$), a *privacy parameter* k enables us to obtain the following results: (i) the probability to recover one vote of honest node is bounded by $\sum_{m=k+1}^{2k} \left(\frac{D}{N}\right)^m \cdot \left(\frac{1}{2}\right)^{2k+1-m}$; (ii) the maximum number of votes revealed by dishonest nodes is $2D$; and, (iii) the maximum impact on the output is $(6k+4)D$. Despite the use of richer social graph structures, we succeed to detect the misbehaving users by manipulating verification procedures based on shortest path scheme and routing tables. An experimental evaluation demonstrates that the dishonest coalition never affects the outcome of the poll outside the theoretical bound of $(6k+4)D$.

Key-words: Social networks, Polling protocol, Secret sharing, Privacy.

* Lorraine University & INRIA Nancy – Grand-Est (Bao-Thien.Hoang@inria.fr)
† Lorraine University & INRIA Nancy – Grand-Est (Abdessamad.Imine@inria.fr).

RESEARCH CENTRE
NANCY – GRAND EST

615 rue du Jardin Botanique
CS20101
54603 Villers-lès-Nancy Cedex

Problème du Sondage dans les Réseaux Sociaux

Résumé : Nous abordons le problème de sondage dans les réseaux sociaux où le caractère secret des informations échangées et la réputation de l'utilisateur sont très critiques. En effet, les utilisateurs désirent préserver la confidentialité de leur vote et dissimuler, le cas échéant, leurs mauvais comportements. Des travaux récents [9, 10] ont proposé des protocoles de sondage basés sur la partage de secret et nécessitant aucune infrastructure cryptographique. Néanmoins, ces protocoles ne sont applicables que si le graphe social a une structure d'anneau et le nombre d'utilisateurs est un carré parfait. En conséquence, l'élaboration de protocoles de sondage indépendamment de ces contraintes reste un problème ouvert.

Dans ce rapport, nous proposons un protocole décentralisé de sondage qui s'appuie sur l'état réel des graphes sociaux. Plus précisément, nous définissons une famille de graphes et montrons que leurs structures constituent la condition nécessaire et suffisante pour assurer la confidentialité du vote et limiter l'impact des utilisateurs malhonnêtes sur la précision de la sortie du scrutin. Dans un système de N utilisateurs, tel que $D \leq N/5$ sont malhonnêtes (et similairement aux travaux [9, 10] qui se limitaient à $D < \sqrt{N}$)), un paramètre de confidentialité k nous permet d'obtenir les résultats suivants : (i) la probabilité de récupérer la voix d'un noeud honnête est bornée par $\sum_{m=k+1}^{2k} \left(\frac{D}{N}\right)^m . \left(\frac{1}{2}\right)^{2k+1-m}$; (ii) le nombre maximum de votes révélés par des nœuds malhonnêtes est de $2D$; et, (iii) l'impact maximum sur la sortie est $(6k+4)D$. Malgré l'utilisation de riches structures de graphes sociaux, nous sommes parvenus à détecter les comportements des utilisateurs malhonnêtes en manipulant des procédures de vérification s'appuyant sur le calcul du plus court chemin et des tables de routage. Une évaluation expérimentale montre que l'influence de la coalition d'utilisateurs malhonnêtes sur le résultat du scrutin ne dépassera pas la borne théorique de $(6k + 4)D$.

Mots-clés : Réseaux sociaux, Protocole de sondage, Partage de secret, Vie privée

Contents

1 Introduction

We have seen the power of social media and its effect on society in the last few years. Online social network (OSN) has been the most useful and typical technology of that. The user number of such networks is blowing up exponentially. Just to demonstrate one typical example, as of now Facebook has more than 901 million active users and 526 million daily active users on average.[1] OSN allows participant to do anything for a variety of purposes concerning business, entertainment, world's events and culture such as getting friendship, publishing and sharing information, exchanging documents, expressing opinions in politics.

In this work, we approach to one of the current practical, useful but sensitive topic in Online Social Networks (OSN), the *polling* process. In general, polling is the way to determine the most favorite choice amongst some options from the participants. Each participant can distribute his preference by submitting vote, and after aggregating all votes, the majority option will be chosen as the final result. For instance, one company of mobile phone has just launched a new product and may want to ask customers whether or not its features are comfortable, and user will choose one option between "Yes" or "No". We here consider simply a binary polling with only two options "+1" or "-1" for the concerning question.

The main objective in such a polling protocol is performing a secure and accurate process to sum up the initial votes with the presence of dishonest users, who try to bias the final result and reveal the votes of honest ones. The polling problem is simple but it takes an important role in incorporating user's opinion online. Thus, currently, there are some studies and solutions for this problem in two approaches, centralized and distributed networks. In the centralized OSN, a central server is used to collect the users' votes and sum up all values to obtain output. Facebook Pool[2] and Doodle[3] are well illustrative examples. However, this approach suffers from server failures and particularly privacy problems: it is not guaranteed the central server will not bias and disclose the user votes.

In our work, we are interested in polling protocol based on decentralized OSN, where privacy of user is improved as information is not concentrated in one place. Recently, Guerraoui et al. [9, 10] proposed, DPol, a simple decentralized polling protocol based on secret sharing scheme (without using cryptography) where both honest and dishonest participants are considered. In DPol, participants care about their reputation. Indeed, they do not want their votes to be disclosed nor their misbehaviors, if any, to be publicly exposed. To dissuade user misbehaviors, distributed verification procedures are manipulated to detect with a non-zero probability these misbehaviors and enable honest users to tag profile of dishonest ones. Moreover, DPol ensures privacy of votes and final result accuracy by limiting the impact of dishonest users. However, DPol has practically some disadvantages. Firstly, DPol relies on a structured overlay, clustering-based structure, which is on top and really apart from the normal social graph. It does not take into account the social links among users in the sense that it builds the uniform distribution of users into groups. This is not practical as we have to target a special case using notion of group instead of reserving the normal structure of the graph. This construction would be necessarily based on centralized solution. Hence we lose the benefit of a fully decentralized polling protocol. Second, the number of users should be a perfect square number such that one graph with N users are divided into $\sqrt{N}$ groups of size $\sqrt{N}$. It should be noted designing decentralized polling protocol without cryptography and constraints (overlay structure and perfect square number of users) imposed in [9, 10] remains a challenge problem.

Contributions. Our objective is to keep the natural property of the graph in the sense user and

[1] http://newsroom.fb.com/content/default.aspx?NewsAreaId=22
[2] http://apps.facebook.com/opinionpolls/
[3] http://www.doodle.com/

social links should be preserved, and each individual can perform the voting process privately and securely without resorting to the group division.

Inspired from [9, 10], first, we propose a design of a simple decentralized polling protocol that uses social graphs. Second, we describe properties required for the social graph to ensure the correctness of the protocol. Furthermore, we cover a general case for the graph topology on which the protocol can run properly. Despite the use of richer social graph structures, one node can receive/send so many duplicated messages from/to other nodes. This can lead to flooding the local storage. By thoroughly using the graph structure, our protocol enables one node to deal with only necessary messages. Instead of accepting all messages, a node stores only the ones passed by the optimal paths. To prevent user misbehaviours, we introduce verification procedures based on shortest path scheme and routing tables. Using the same notion of privacy parameter k in [9, 10], we get the following results in a system of size N with D dishonest users: one vote of honest node is recovered with the probability at most $\sum_{m=k+1}^{2k} \left(\frac{D}{N}\right)^m . \left(\frac{1}{2}\right)^{2k+1-m}$; and up to $2D$ votes can be revealed by the dishonest coalition, and the impact from the dishonest coalition to the final result is at most $(6k+4)D$. We validate our solution with a performance evaluation which shows that our protocol is accurate and close to the theoretical average impact, that is $4k + 2\alpha + 2$, where α is the proportion of users correctly voting. Our result encourages the use of polling protocol without transforming the social graphs into other overlay structures.

Outline. This paper is organized as follows. Section 2 describes our polling model, and introduces a family of social graphs. Section 3 presents our polling protocol with its correctness properties. We establish formally the relation between the protocol and the family of social graphs, and analyse different complexities to perform the polling in two cases, presence or not presence of dishonest nodes, in Section 4 and Section 5 respectively. Section 6 illustrates our experimental results. We review related work in Section 7 and conclude the paper with future research in Section 8.

2 Polling and Social Network Models

This section introduces what are ingredients of polling models and presents the graph models to describe social networks. It should be noted that we consider the same assumptions like the work [9, 10].

2.1 Polling Model

The polling problem consists of a system with N uniquely identified nodes representing users of a social network. Each participant u_n (or simply, n) expresses its opinion by giving a vote $v_n \in \{-1, 1\}$. After collecting the votes of all nodes, the expected outcome is $\sum_n v_n$. In this work, we consider the following assumptions:

Each node is able to communicate with its neighbors (e.g., direct friends), and is either honest or dishonest. The honest node completely complies with the protocol and takes care about its privacy and reputation in the sense that the vote value is not disclosed. All dishonest nodes can form a coalition to get the full knowledge of the network and try to do everything to achieve these goals without being detected: (i) bias the result of the election by promoting their votes or changing the values they received from other honest nodes; (ii) infer the opinions of other nodes. However, they also want to protect their reputation from being affected. In order to unify the opinions and not give compensating effects, all dishonest nodes make the single coalition $\mathcal{D}$ of size D. Nevertheless, they are still selfish in the sense that each dishonest node prefers to take care about its own reputation to covering up each other.

In order to prevent and reduce the incorrect behaviors, there is an activity affected to profile of concerned node. In particular, if node u is detected as misbehavior one by v then u's profile is tagged with statement "u has been detected bad behavior by v" and in v's profile has statement "u is bad guy". Furthermore, we do not take into account the situation that dishonest nodes wrongfully blame honest ones, or do Sybil attacks and spam since that kinds of misbehavior can be detected by some tools or several existing systems such as SybilLGuard[20], SybilLimit [19], [14, 18] (for filtering wrongful blames), and [14, 17] (for mitigating spam).

2.2 Social Network as a Graph Models

We present the social network in our problem as the form of models of social graph. In this section, firstly, we define the terms and notations of graph used throughout our work. Later, we demonstrate the family of graphs including the ideal case (network without dishonest nodes) and normal case (network with the presence of dishonest nodes).

Notations. Let $G = (V, E)$ be an undirected graph where $V = \{u_0, u_1, ..., u_{N-1}\}$ is a set of uniquely identified nodes of size N, and E is an edge set. Each node is either honest or dishonest. We represent $\mathcal{H}(X)$ and $\mathcal{D}(X)$ as the set of honest nodes and dishonest nodes of size $D = |\mathcal{D}(\mathcal{X})|$ in graph X. For a node $u_n \in V$, let us identify the following notations: d_n as a degree (a number of neighbors) of u_n; $\mathcal{R}(u_n)$ (or simply, $\mathcal{R}_n$) as the set of neighbors of u_n; $\mathcal{F}_n$ and $\mathcal{Q}_n$ ($\mathcal{F}_n, \mathcal{Q}_n \subseteq \mathcal{R}_n$) are respectively set of neighbors that u_n sends and receives messages.

Paths and Distances. Given two nodes $u, v \in V$, they can connect directly or not. We denote by function $e(u, v)$ this kind of relation, namely, $e(u, v) = 1$ if there is a link between u and v, otherwise $e(u, v) = 0$.

A path p of length $l \in \mathbb{N}$ in the graph is an ordered sequence of $l + 1$ nodes such that there exists an edge connecting two consecutive nodes in the sequence: $p = \langle u_{k_1}, u_{k_2}, \ldots, u_{k_{l+1}} \rangle$ with $u_{k_i} \in V$, $e(u_{k_i}, u_{k_{i+1}}) = 1$, $1 \leq i \leq l$. We write $l(p)$ to refer the length of path p, i.e., number of the edges of p. As $e(u_{k_i}, u_{k_{i+1}}) = 1$ then $l(\langle u_{k_i}, u_{k_{i+1}} \rangle) = 1$. If path p contains only one node, $l(p) = 0$.

For two nodes $u, v \in V$, let $p(u, v)$ be a path connecting between u and v and $Pa(u, v)$ be the set of all such paths. We write $x \in p(u, v)$ if path $p(u, v)$ contains node x. For two paths $p(u_1, v_1)$, $p(u_2, v_2)$, we define the intersection of them as follows: $p(u_1, v_1) \cap p(u_2, v_2) = \{x \in V \mid x \in p(u_1, v_1) \text{ and } x \in p(u_2, v_2)\}$.

Additionally, each node is either honest or dishonest. Thus, to transmit messages between two nodes u and v, it is important to consider the honesty property of each node (i.e., checking whether node is honest or dishonest) in the paths connecting them. Particularly, if u and v are directly connected, i.e., $e(u, v) = 1$, we should investigate the honesty property of u and v. The transmission is secure only if they are all honest and is unsecured in other case. If $e(u, v) = 0$, we should examine all paths connecting between u and v. For a path $p(u, v) = \langle u \equiv u_{k_1}, u_{k_2}, ..., u_{k_m} \equiv v \rangle$ (where $e(u_{k_i}, u_{k_{i+1}}) = 1$, $1 \leq i \leq m - 1$), we have to check honest property of each intermediate node u_{k_i}. The transmission in that path is secure only if all nodes are honest and we call it "honest path". If there exists at least one honest path between u and v, it guarantees the correct information from u (or v) will approach to v (or u).

We describe, more formally, the way to check the secure transmission between nodes u and v by using concept "trust level". Let us firstly define the value q for two directly connected nodes u, v as $q(u, v) = 1$ if u, v are honest, and $q(u, v) = 0$ otherwise. Note that the value of $q(u, u)$ depends on whether u is honest or not. The "trust level" for a specific path p is:

$$T(p) = \begin{cases} q(u, u) & \text{if } p = \langle u \rangle \\ q(u_1, u_2).T(\langle u_2, ..., u_m \rangle) & \text{if } p = \langle u_1, u_2, ..., u_m \rangle \end{cases} \quad (1)$$

Here, notation "." is multiplication operation.

Now for all paths $Pa(u, v)$ linking two nodes u and v, the trust level $T(u, v)$ is given as:

$$T(u, v) = \sum_{p \in Pa(u,v)} T(p) \tag{2}$$

Obviously, $T(u, v) = 0$ if u or v is dishonest. Therefore, we often use $T(u, v)$ when u, v are honest. In case that transmission in path $p = \langle u_1, u_2, ..., u_m \rangle$ is safe, i.e., $T(p) > 0$, then p is a "honest path". For instance, consider the graph given in Figure 1, where we consider dishonest nodes are w and y and the remaining are honest ones. According to Formula 2: (i) $T(w, w) = 0$ as $q(w, w) = 0$, and (ii) $T(s, d) = T(p(\langle s, w, d \rangle)) + T(p(\langle s, x, d \rangle)) + T(p(\langle s, x, y, d \rangle)) + T(p(\langle s, x, y, t, d \rangle)) = 1$ since $q(s, w) = 0$ and $q(x, y) = 0$.

For a graph G, there exists, for all pairs of honest nodes u, v, at least one honest path between them, then G is called "honest graph". Formally, G is honest if $\forall u, v \in \mathcal{H}(G) : T(u, v) > 0$.

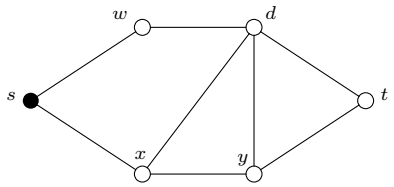


Figure 1: Example of social graph

Shortest paths. We illustrate by $p_S(u, v)$ and $Pa_S(u, v)$ the shortest path and the set of all shortest paths between two nodes u and v. In [8], a simple but fast and accurate algorithm for the approximation of shortest paths between pair of nodes in the real-world graph is presented. We can use this method to determine the shortest paths and distances between two nodes. The length of the shortest path between u and v, is denoted by $\delta(u, v)$, i.e., $\delta(u, v) = l(p_S(u, v))$.

Graph Model. So far, we presented the polling model in which participants are either honest or dishonest. Like [9, 10], we use a predefined parameter $k \in \mathbb{N}$ (this parameter will be detailed in section 3.1) to present the features of our social graphs. Let $G = (V, E)$ be a social graph with the following properties:

Property 1 (P_{g_1})**.** *$d_n \geq 2k + 1$ and $|\mathcal{F}_n| = |\mathcal{Q}_n| = 2k + 1$, for every $u_n \in V$.*

Property 2 (P_{g_2})**.** *G is a honest graph, i.e., for every honest nodes u, v, there exists a path $p(u, v)$ containing only intermediate honest nodes.*

Property 3 (P_{g_3})**.** *$D < N/2$.*

From these properties, we characterize two families of graphs:

(i) $\mathcal{G}_1 = \{G \mid \mathcal{D}(G) = \varnothing$ and G satisfies $P_{g_1}\}$.

(ii) $\mathcal{G}_2 = \{G \mid \mathcal{D}(G) \neq \varnothing$ and G satisfies P_{g_1}, P_{g_2} and $P_{g_3}\}$.

Graphs in $\mathcal{G}_1$ contain no dishonest nodes and in $\mathcal{G}_2$ are normal graphs with the existence of dishonest nodes. According to Property P_{g_1}, each node has a set of receivers ($\mathcal{F}_n$) and a set of senders ($\mathcal{Q}_n$) to establish communication and they have the same size and may be disjoint. Property P_{g_2} ensures each honest node always obtains one correct version of data from other honest ones. Property P_{g_3} enables us to limit the control of dishonest users in the whole system.

Algorithm 1: POLLING ALGORITHM AT NODE u_n, $n \in \{0, 1, ..., N-1\}$

Input:
v_n: A vote of node, value in $\{-1, 1\}$
d_n: degree of node
k : privacy parameter
$\mathcal{R}_n$: set of direct neighbors
$\mathcal{F}_n$: set of neighbors to send shares
$\mathcal{Q}_n$: set of neighbors to receive shares

Variables:
c_n: collected data, $c_n = 0$
$\mathcal{C}_n$: set of possible collected data
$\mathcal{C}_n[\{0, 1, ..., N-1\} \rightarrow \varnothing]$
h_n: set of final choosing collected data
$h_n[\{0, 1, ..., N-1\} \rightarrow \perp]$
Γ_n: routing table
$\Gamma_n[\{0, 1, ..., N-1\} \rightarrow \varnothing]$

Output: result

Algorithm
1 Share(v_n, $\mathcal{F}_n$) | ReceiveShareEvent
2 Broadcast(n, c_n, 1, $\mathcal{R}_n$) | ReceiveDataEvent
3 Aggregate()

Procedure Share(v_n, $\mathcal{F}_n$)
4 $\mathcal{P}_n \leftarrow \varnothing$
5 **for** $i \leftarrow 1$ **to** k **do**
6 $\mathcal{P}_n \leftarrow \mathcal{P}_n \cup \{v_n\} \cup \{-v_n\}$
7 **end**
8 $\mathcal{P}_n \leftarrow \mathcal{P}_n \cup \{v_n\}$
9 $\mu_n \leftarrow_{rand} \mathcal{P}_n$
10 **for** $i \leftarrow 0$ **to** $2k$ **do**
11 send(SHARE, $\mu_n[i], \mathcal{F}_n[i]$)
12 **end**

Procedure ReceiveShareEvent(SHARE, p, r)
13 **if** $(r \in \mathcal{Q}_n \wedge p \in \{-1, 1\})$ **then**
14 $c_n \leftarrow c_n + p$
15 **end**

Procedure Broadcast(n, c_n, l_n, $\mathcal{R}_n$)
16 **foreach** $(r \in \mathcal{R}_n)$ **do**
17 send(DATA, n, c_n, l_n, r)
18 **end**

Procedure ReceiveDataEvent(DATA, s, c_s, l_s) from neighbor identity t
19 **if** $(s = n$ or $l_s > \delta_L(s, n))$ **then** exit
20 **if** $(c_s \notin \mathcal{C}_n[s])$ **then**
21 $\nu_s \leftarrow c_s$
22 $\mathcal{C}_n[s] \leftarrow \mathcal{C}_n[s] \cup \{c_s\}$
23 Broadcast(s, ν_s, $l_s + 1$, $\mathcal{R}_n \setminus \{t\}$)
24 **else**
25 $\nu_s \leftarrow \perp$
26 **end**
27 $\Gamma_n[s] \leftarrow \Gamma_n[s] \cup \{(t, c_s, \nu_s, l_s)\}$

Procedure Aggregate()
28 result $\leftarrow 0$
29 **for** $s \leftarrow 0$ **to** $N-1$ **do**
30 **if** $(s \neq n)$ **then**
31 $h_n[s] \leftarrow$ CheckInconsistency(s)
32 **else**
33 $h_n[s] \leftarrow c_n$
34 **end**
35 result $\leftarrow$ result $+ h_n[s]$
36 **end**

Procedure CheckInconsistency(s)
37 **if** $(|\mathcal{C}_n[s]| = 1)$ **then**
38 **return** $\mathcal{C}_n[s][0]$
39 **else**
40 **return** *correct value after verifying* $\Gamma[s]$ *of neighbors*
41 **end**

3 Protocol

In this section, we first present our polling protocol and give some properties of this protocol. We assume there is no crash and message loss.

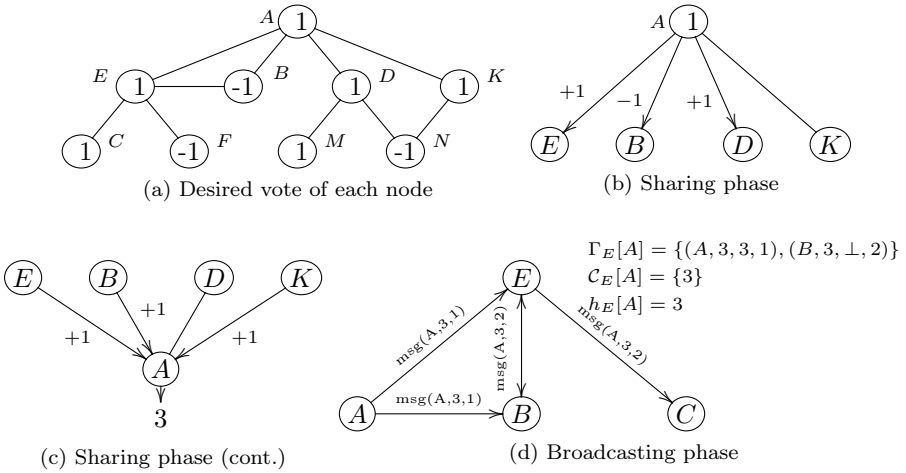
3.1 Description

Generally, the polling protocol includes three phases (see Algorithm 1): (*i*) *Sharing*, (*ii*) *Broadcasting* and (*iii*) *Aggregating*. Phase *Sharing* describes the generation, distribution of a set of shares of each node to its neighbors as well as collecting these shares from its neighbors. In the *Broadcasting* phase, each node broadcasts messages containing the total shares, which are collected in the Sharing phase, to its direct and indirect neighbors. The last phase, *Aggregating*, shows the process that each node decides data received from other nodes and computes the final outcome.

Sharing. In this phase, each node u_n contributes its opinion by sending a set of shares expressing its vote $v_n \in \{-1, 1\}$ to its neighbors. We inspired the sharing scheme proposed in [5] to generate shares. Namely, u_n generates $2k+1$ shares $\mathcal{P}_n = \{p_1, p_2, ..., p_{2k+1}\}$ where $p_i \in \{-1, 1\}$, $i = 1, 2, ..., 2k+1$ including: $k+1$ shares of value v_n, and k shares of opposite v_n's value. Later it generates randomly a permutation of $\mathcal{P}_n$, and sends these $2k+1$ messages to $2k+1$

direct neighbors. Lines 4–12 in Algorithm 1 describe this phase. Node also receives exactly $2k + 1$ messages from its direct neighbors. We can target this strict number of nodes in the set of receivers and senders by the following approach: each node u_n will determine all possible receiver sets $\mathcal{F}_n \subseteq \mathcal{R}_n$ such that $|\mathcal{F}_n| = 2k+1$ by choosing $2k+1$ arbitrary elements from $\mathcal{R}_n$, i.e., $\mathcal{F}_n = \{m_{n_1}, m_{n_2}, ..., m_{n_{2k+1}}\}$, $\forall m_{n_i} \in \mathcal{R}_n$. Node u_n also knows all other possible set $\mathcal{F}_p$ of any other node u_p. From this it can identify all possible tuples of N sets of the form $(\mathcal{F}_0, \mathcal{F}_1, ..., \mathcal{F}_{N-1})$. For each tuple $(\mathcal{F}_0, \mathcal{F}_1, ..., \mathcal{F}_{N-1})$, it will check whether the following condition is satisfied: each element $m_{n_i} \in \mathcal{F}_n$, $0 \leq n < N$ must belong to exactly other $2k$ sets $\mathcal{F}_{i_1}$, $\mathcal{F}_{i_2}$, ..., $\mathcal{F}_{i_{2k}}$, $n \neq i_j$, $j = 1, 2, ..., 2k$. If there exists a tuple $(\mathcal{F}_0, \mathcal{F}_1, ..., \mathcal{F}_{N-1})$ fulfilled above conditions, the requirement about number of receivers and senders at each node will be satisfied. For instance, specially, each node u_n defines the set of receivers $\mathcal{F}_n = \{u_{(n+1) \bmod N}, u_{(n+2) \bmod N}, \ldots, u_{(n+2k+1) \bmod N}\}$ of size $2k + 1$. Besides, wee see that $\mathcal{Q}_n = \{u_{(n-1) \bmod N}, u_{(n-2) \bmod N}, \ldots, u_{(n-2k-1) \bmod N}\}$ of size $2k + 1$.

After all nodes collect $2k + 1$ shares from its neighbors, and sums into *collected data* c_n (lines 13–15 in Algorithm 1), this phase is complete. Figures 2 illustrates an example of the protocol for $k = 1$. Figure 2a presents desired vote of each node, whereas Figure 2b depicts the sharing phase at node A. Node A would like to vote $+1$, thus, it generates a set of $2k + 1 = 3$ shares $\{+1, -1, +1\}$ which total equals to $v_A = 1$. Figure 2c shows node A collects the shares from its neighbors and computes the collected data $c_A = 3$.



(a) Desired vote of each node (b) Sharing phase

(c) Sharing phase (cont.) (d) Broadcasting phase

Figure 2: Polling algorithm for $k = 1$

Broadcasting. In this phase, each node u_n encapsulates the collected data c_n with its identity n and length counter l_n, which expresses the length of the path message has passed (initially, $l_n = 1$), into message msg and disseminates it to all neighbors (lines 2 and 16–18 in Algorithm 1). This action is depicted in Figure 2d. When u_n receives from u_t message $msg(s, c_s, l_s)$ emitted from the source u_s, it performs the following actions (see ReceiveDataEvent() in Algorithm 1):

1. *Loop detection*: u_n checks contents of msg and detects the loop based on the source's identity (line 19 in Algorithm 1). If this message is the one u_n has emitted earlier, i.e., $s = n$, then u_n simply drops the message and does not need to inform u_t. Otherwise, u_n accepts msg.

2. *Message Forwarding*: For a message passing the loop detection, u_n should get data c_s and forward to its friends except u_t.

   We see that, naively approaching, u_n can receive c_s from many disjoint paths (without loop) connecting between u_s and u_n. However, the number of paths can be blown up to exponential value. More specifically, the worst case is when G is clique and one node has $N-1$ friends. Each message passes through all nodes in the network, and thus, the number of possible paths between u_s and u_n is $(N-1)(N-2)...1 = (N-1)!$. This motivates us to find out an optimal solution to bound the number of messages emitted from u_s that u_n should receive without losing any necessary information.

   Instead of using naive approach, we propose other technique which is small but very useful and much more optimal: node receives messages which passed by paths with the limited length rather than accepting all. Here, for messages broadcasting from u_s, we use bread-first expansion with the assumption that transmitting message in one edge takes one time unit. Hence, we see that firstly u_n receives messages from u_s in the shortest path $p_S(u_s, u_n)$, and then from other paths of greater length. By the way, the content of messages can be changed by some intermediate dishonest nodes in the path $p \in Pa(u_s, u_n)$. Thus, we should take care the intermediate nodes. For each intermediate node x, it receives message from u_s in $p_S(u_s, x)$ first and from the longer path later. Node u_n also receives message, which passed x, from the shortest path $p_S(x, u_n)$ first and then from other longer paths $p(x, u_n)$. Therefore, u_n receives messages, which are broadcast from u_s and passed x, from the paths with length $\delta(u_s, x) + \delta(x, u_n)$ first, and from other longer paths later. To take care of all possible changes in contents, u_n should receive all messages which already passed all intermediate nodes. And so, the maximum length of the paths passing message u_n should receive is $max_x\{\delta(u_s, x) + \delta(x, u_n)\}$.[4]. In case that for all node x, $p_S(u_s, x)$ and $p_S(x, u_n)$ have some common nodes (different from x), u_n should not receive messages from the paths of length $\delta(u_s, x) + \delta(x, u_n)$ since they have a loop inside. It should receive messages from paths of length $\delta(u_s, u_n)$ instead. So, we combine all of these results, and define one value which u_n (resp. u_s) could use to determine the maximum length of paths which deliver messages from u_s (resp. u_n) to u_n (resp. u_s) as follows:

$$\delta_L(u_s, u_n) = \begin{cases} \max_{x \in U_{sn}}\{\delta(u_s, x) + \delta(x, u_n)\} & \text{if } u_s \neq u_n \wedge |U_{sn}| > 0 \\ \delta(u_s, u_n) & \text{otherwise} \end{cases} \quad (3)$$

   where $U_{sn} = \big\{x \in V | x \neq u_s, u_n \text{ and } \exists p_1 \in Pa_S(u_s, x), p_2 \in Pa_S(x, u_n) \text{ s.t. } p_1 \cap p_2 = \{x\}\big\}$.

   For message with $l_s \in \big[\delta(u_s, u_n), \delta_L(u_s, u_n)\big]$, node u_n accepts and does the following activities, otherwise it simply eliminates that message. Line 19 in Algorithm 1 shows this verification.

   The activities in the case $l_s \in \big[\delta(u_s, u_n), \delta_L(u_s, u_n)\big]$ are as follows (lines 20–27 in Algorithm 1): u_n checks $\mathcal{C}_n[s]$, a set of possible values emitted from the source with identity s, to determine whether c_s is already presented in it. If c_s is not stored in $\mathcal{C}_n[s]$, u_n will add it into $\mathcal{C}_n[s]$, and then forward message $msg(s, \nu_s, l_s + 1)$, where ν_s is value to be sent (in this case $\nu_s = c_s$), to other direct neighbors except u_t. All information about the messages from source u_s is stored in the *routing table* $\Gamma_n[s]$ which is used for checking inconsistency later. This table contains the following fields: first field is neighbor identity from which it received message (e.g., t), second one is the receiving value (e.g., c_s), third one is the value to be forwarded (e.g., ν_s), and last field is the length of the path passing message

[4] See Lemma 5 for the correctness of this consideration.

from the source u_s (i.e., l_s). In this case, u_n adds tuple (t, c_s, ν_s, l_s) into $\Gamma_n[s]$. In other case that $\mathcal{C}_n[s]$ has value c_s inside, u_n does not need replicating that value in $\mathcal{C}_n[s]$, as well as forwarding it to other friends as it already did earlier. It just stores information in the routing table, by setting the sent value as null, i.e., $\nu_s = \perp$ (null).

Figure 2d depicts the process when node E receives message emitted from A. When $msg(A, 3, 1)$ with length 1 arrives to E, it stores $c_A = 3$ into set of possible collected data of source A, that is $\mathcal{C}_E[A]$. It then forwards $msg(A, 3, 2)$ with length 2 to B and C and adds a tuple $(A, 3, 3, 1)$ into routing table of source A, i.e., $\Gamma_E[A]$. Notice that A also sends $msg(A, 3, 1)$ to B with the same length as the one to E, thus, B gets the same message and does the same actions like E. Node E gets forwarded message with length 2 from B. Since that is the second message having the same source and collected data, but higher length, E does not forward it. Node E just inserts one more tuple $(B, 3, \perp, 2)$ expressing the information received from B into routing table $\Gamma_E[A]$.

Once there is no broadcasting messages in the network, this phase is over. Since each node just sends and receives a finite number of messages, and all messages eventually arrives, it is guaranteed this phase terminates correctly.

Aggregating. In this phase, u_n has to decide the collected data of other nodes before calculating the final result. To make decision for node u_s, it checks $|\mathcal{C}_n[s]|$ (lines 37–41 in Algorithm 1): if $|\mathcal{C}_n[s]| = 1$, the single element in $\mathcal{C}_n[s]$ is chosen as a correct collected data, otherwise there exists an inconsistency and it should do the verification: requesting all routing tables $\Gamma[s]$ of neighbors and indirect neighbors to check information received and forwarded by them. If one node is detected that it already sent different values of its data or its receiving information, then an alarm is raised and that node is tagged in its profile. By doing this, u_n also gets the correct collected data of source u_s.[5] So, in any case, u_n achieves the correct copy of collected data of source u_s. It then stores that value as one item $h_n[s]$ in the array h_n, which contains collected data of other nodes, and adds into result (lines 29–36 in Algorithm 1). After checking and summing up all collected data of nodes (including its own collected data c_n), u_n obtains the final result (that is $\mathsf{result} = c_n + \sum_{i \neq n} h_n[i]$).
For instance, we consider Figure 2d again. From formula (3), we see that $\delta_L(A, E) = \delta(A, B) + \delta(B, E) = 2$. After receiving message $msg(A, 3, 2)$ from node B, and updating routing table, node E makes final decision to choose value from source A. As the set $\mathcal{C}_E[A]$ is singleton, it will set $h[A] = \mathcal{C}_E[A][0] = 3$. This value will be used to compute final outcome of polling later.

3.2 Properties of protocol

In section 2.1, we already introduced the characteristics of the polling model. It implies that our protocol should have some properties such that the system can run correctly with (or without) the existence of dishonest nodes. Namely, each honest node outputs the correct polling result, controls the impact from the dishonest nodes, and not disclose its private information, whereas the dishonest coalition could not control the polling process or fool an entire network without being detected. In this section, we clarify those desirable properties by stating what protocol should achieve with (or without) the existence of dishonest nodes such as accuracy and privacy.

We say the sharing vector μ_v (or μ_n) of vote $v \in \{-1, 1\}$ (of node n) is the subset of multiset of shares generated from vote v (or node n), i.e., $\mu_v \in 2^{\mathcal{P}_v}$ (or $\mu_n \in 2^{\mathcal{P}_n}$) where $\mathcal{P}_v = \mathcal{P}_n = \{p_1, p_2, ..., p_{2k+1}\}$ and $p_i \in \{-1, 1\}$. We use $\mu_v(\mathcal{D})$, or simply $\mu(\mathcal{D})$, to mention the shares generated from vote v that coalition $\mathcal{D}$ receives. We present $\mu_v[x]$ and $\mathcal{P}_v[x]$ as the number of value x in sharing vector μ_v and set $\mathcal{P}_v$, respectively, i.e., $\mu_v[x] = |\{t | t \in \mu_v \text{ and } t = x\}|$, $\mathcal{P}_v[x] = |\{t | t \in \mathcal{P}_v \text{ and } t = x\}|$.

[5] See Lemma 8 for the detail of this verification.

We denote the sum of all elements in the sharing vector μ by $g(\mu)$, i.e., $g(\mu) = \sum_i x_i$ where $x_i \in \mu$. Moreover, to express that node u (or coalition $\mathcal{D}$, resp.) reveals vote x, we use the notation $u \vDash x$ (or $\mathcal{D} \vDash x$, resp.).

Privacy. The privacy property expresses the ability of the system to prevent the private information from being leaked to the dishonest nodes. In other words, the coalition could not reveal any information of particular honest node beyond what it can deduce from its own vote, the output of computation and the shares of votes.

Definition 1 (Privacy)**.** *The protocol is said* private *if the dishonest nodes cannot learn anything about the vote of honest node. More formally, for any honest node u_n with vote v_n, there exists a negligible function $\xi(k)$ such that:*

$$Pr[\mathcal{D} \vDash v_n] \leq \xi(k) \tag{4}$$

Accuracy. We define the impact of dishonest nodes as the difference between the output and the expected result. In our case, vote is either "+1" or "-1", and thus, with the system of N nodes, the maximum and minimum final results are N and $-N$ respectively. This implies the maximum difference amongst the final outputs is $2N$. As defined in [7], accuracy is given by the maximum impact with respect to the maximum difference of the final outputs:

$$\Lambda = \frac{1}{2N} \cdot \max_{n \in \mathcal{H}(G)} \Delta(\mathsf{result}_n, \sum_{i=0}^{N-1} v_i) \tag{5}$$

where result_n is the output of the poll (see Algorithm 1). Here and throughout this work, we denote by $\Delta(x, y)$ the difference between value x and y, i.e., $\Delta(x, y) = |x - y|$.

Definition 2 (Accuracy)**.** *The protocol is said* accurate *if there exists a negligible function $\xi(k)$ such that $\Lambda \leq \xi(k)$.*

4 Protocol and graph without dishonest nodes

In this section, we consider only graphs of family $\mathcal{G}_1$ and analyze the correctness (including accuracy and termination) of our protocol when deployed with graphs of $\mathcal{G}_1$. Next we give spatial, message and time complexities. Finally, we show properties of $\mathcal{G}_1$ are necessary and sufficient condition to ensure the correctness of our protocol. Here we do not consider the privacy property as there is no dishonest nodes in this case.

Lemma 1 (Accuracy)**.** *After following the polling protocol, each node gets the accurate expected output.*

Proof. We will prove that the output of the protocol at each node is $\mathsf{result} = \sum_{i=0}^{N-1} v_i$.

In the sharing phase, each node u_n sends a sharing vector $\mu_n = \mathcal{P}_n = \{p_{n_1}, p_{n_2}, ..., p_{n_{2k+1}}\}$ to its $2k+1$ direct neighbors and $g(\mu) = (k+1).v_n + k.(-v_n) = v_n$. In addition, u_n receives a set of shares $\{p'_{n_1}, p'_{n_2}, ..., p'_{n_{2k+1}}\}$ from its $2k+1$ direct neighbors and gets the collected data with value of $c_n = \sum_{j=1}^{2k+1} p'_{n_j}$. With the assumption that there is no dishonest node and without crash or message loss, each message from the source successfully reaches the destination, and thus the set of all sending shares of all nodes will be exactly coincided with the set of all receiving shares of all nodes, namely:

$$\bigcup_V \{p_{n_1}, p_{n_2}, ..., p_{n_{2k+1}}\} = \bigcup_V \{p'_{n_1}, p'_{n_2}, ..., p'_{n_{2k+1}}\}$$

In the broadcasting phase, each messages msg from each node is broadcast or forwarded, and finally arrived to all direct and indirect neighbors. It infers that the array h_n contains all collected data of all nodes in the system and these values comes from all the receiving shares of all the nodes. Consequently, the final computation gives us the value:

$$\mathsf{result} = \sum_{\substack{0 \le i < N \\ i \neq n}} h_n[i] + c_n = \sum_{i=0}^{N-1} c_i = \sum_{i=0}^{N-1} \sum_{j=1}^{2k+1} p'_{i_j} = \sum_{i=0}^{N-1} \sum_{j=1}^{2k+1} p_{i_j} = \sum_{i=0}^{N-1} v_i$$

Lemma 2 (Termination)**.** *The polling protocol is guaranteed to terminate.*

Proof. In the sharing phase, each node has to send and receive a finite number of of messages, that is $2k+1$. In the broadcasting phase, each node u_n receives and forwards the finite number of message from source u_s, because it just receives message passed through the path with length in the interval $\big[\delta(u_s, u_n), \delta_L(u_s, u_n)\big]$, not all message departing from s. Moreover, as there is no looping, no losing message, no crash, each phase terminates correctly. The algorithm has a finite phase, it follows the protocol finally terminates.

Proposition 1 (Spatial complexity)**.** *The total space each node must hold is $O(N.d_n)$.*

Proof. Each node u_n needs to maintain a list of d_n direct neighbors to contact, a set $\mathcal{C}_n$ to get a set possible collected data of other nodes, a routing table Γ_n, a set h_n to store the final choosing collected data which size is $N-1$. As there is no dishonest node, node u_n just inserts into $\mathcal{C}_n[s]$ one value emitted from source u_s through the shortest path between u_s and u_n, i.e., $|\mathcal{C}_n[s]| = 1$, and thus $|\mathcal{C}_n| = N-1$. The worst case for the routing table $\Gamma_n[s]$ is when all shortest paths between u_s and u_n are passed by its friends. In other words, $\Gamma_n[s]$ contains at most d_n rows. It infers that $|\Gamma_n| \le (N-1).d_n$. Therefore, the spatial complexity in this case is $O(d_n) + O(N-1) + O((N-1).d_n) = O(N.d_n)$.

Proposition 2 (Message complexity)**.** *The number of messages in our protocol is $O(N.d_n + k)$.*

Proof. In the sharing phase, node u_n sends $2k+1$ messages to its direct neighbors. In the broadcasting phase, it sends d_n messages containing collected data to all of its direct neighbors. It also takes a role as an intermediate node by forwarding message to all of its neighbors except the node it got that message. For each collected data emitted from source u_s, the number of messages node u_n forwards is equal to the number of elements in the set $\mathcal{C}_n[s]$. The cause is that, after receiving the first message of value c_s passed by the shortest path with length $\delta(u_s, u_n)$, u_n stores c_s into $\mathcal{C}_n[s]$, forwards message to $(d_n - 1)$ neighbors, and never does this action for the message having the same value but higher length later. Thus the number of forwarded messages is $|\mathcal{C}_n|.(d_n - 1)$. In ideal situation, with the same explanation as one in the spatial complexity, we have $|\mathcal{C}_n[s]| = 1$, and $|\mathcal{C}_n| = N-1$. Accordingly, the total message complexity is $O(2k+1) + O((N-1).(d_n - 1)) = O(N.d_n + k)$.

Proposition 3 (Time complexity)**.** *Assuming the system is synchronous one in which time generates in round. Then the protocol operates in $O(k + N^2)$ rounds.*

Proof. The sharing phase operates in $2k+1$ rounds. In the broadcasting phase, node u_n broadcasts collected data to d_n neighbors and this takes d_n rounds. As there is no dishonest node in this ideal case, a collected data has to be delivered to some node u_s through the path with length $\delta(u_n, u_s)$. For each intermediate node u_t of this path, it forwards to its $d_t - 1$ friends. It infers that the collected data c_n requires $\max_{0 \le s < N}\{\delta(u_n, u_s).\max_{u_t \in p_S(u_n, u_s)}\{d_t - 1\}\}$ rounds. Therefore, the time complexity is $O(k) + O(\max_{0 \le s < N}\{\delta(u_n, u_s).\max_{u_t \in p_S(u_n, u_s)}\{d_t - 1\}\}) = O(k + (N-1).(N-2)) = O(k + N^2)$.

Theorem 1. *The properties of $\mathcal{G}_1$ are the necessary and sufficient condition for the polling protocol to be deployed correctly in the system without dishonest nodes.*

Proof. The sufficient conditions ($\Leftarrow$), is proved in Lemma 1 and 2. We only examine the remaining of the theorem, the necessary conditions ($\Rightarrow$). Consider a general graph G. We will show $G \in \mathcal{G}_1$. In the sharing phase of the protocol, each node u_n sends (or receives) exactly $2k+1$ messages, i.e., $|\mathcal{F}_n| = |\mathcal{Q}_n| = 2k+1$. It can send (or receive, resp.) messages to (or from, resp.) the same or different neighbors. Thus, $d_n \geq 2k+1$. Notice that if these two sets are disjoint, i.e., $\mathcal{F}_n \cap \mathcal{Q}_n = \varnothing$, then u_n has at least $|\mathcal{F}_n| + |\mathcal{Q}_n| = 2(2k+1)$ neighbors. The set of the graphs with condition $d_n \geq 2k+1$ also includes the set of graphs with condition $d_n \geq 2(2k+1)$. Therefore, to apply protocol correctly, G must have the property P_{g_1}, and we have, $G \in \mathcal{G}_1$.

5 Protocol and graph with dishonest nodes

In this section, we revisit the relation between protocol and graph, but approach it with the presence of D dishonest nodes. We consider graphs of family $\mathcal{G}_2$ and analyze the correctness (including privacy and accuracy) of our protocol when deployed with graphs of $\mathcal{G}_2$. Next we give spatial, message and time complexities. Finally, we show properties of $\mathcal{G}_2$ are necessary and sufficient condition to ensure the correctness of our protocol. Here we do not consider the termination property as it is similar to Lemma 2.

Privacy. We present the privacy of the protocol in Lemmas 3 and 4.

Lemma 3. *In the first phase, a node's vote is revealed by dishonest coalition with certainty if and only if its $k+1$ direct dishonest neighbors received the shares corresponding to its vote.*

Proof. The proof of the direction ($\Leftarrow$) of lemma is trivial. We only need to show the reverse direction ($\Rightarrow$). Assume the contrary, i.e., coalition $\mathcal{D}$ knows vote v_n of one node n but it gets less than $k+1$ shares of value v_n. We dismiss the case that all $2k+1$ direct neighbors of n are dishonest, as otherwise, $\mathcal{D}$ always gets the vote v_n (e.g., by computation $g(\mu) = (k+1).v_n - k.v_n = v_n$). From the viewpoint of $\mathcal{D}$, the best case for them to reveal vote is when it gets $2k$ shares of n. Notice that, $\mathcal{D}$ breaks out the vote if and only if the summing of all shares equals to a vote v_n. This infers the remaining share which $\mathcal{D}$ does not get has value of 0. This contradicts to the value of vote.

Lemma 4 (Privacy preservation)**.** *The probability that dishonest coalition $\mathcal{D}$ reveals vote v of a given honest node n is bounded by $\sum_{m=k+1}^{2k} \left(\frac{D}{N}\right)^m . \left(\frac{1}{2}\right)^{2k+1-m}$ and the protocol is private.*

Proof. Let X and Y be the events "size of $\mu_v(\mathcal{D})$" and "all $k+1$ shares in the sharing vector are identical", respectively.

From Lemma 3, the coalition $\mathcal{D}$ reveals vote v when $\mu_v(\mathcal{D})$ contains $k+1$ identical shares (and their value equals to v_n). Thus, the probability that $\mathcal{D}$ discloses v is equal to the one of event that $\mathcal{D}$ gets $k+1$ elements of v in which $k+1$ ones are the same (we do not consider the case that $\mu(\mathcal{D})$ gets $2k+1$ shares of v). Formally, $Pr[n \in \mathcal{H}(G), \mathcal{D} \vDash v] = Pr(\{X \geq k+1\}.Y) = \sum_{m=k+1}^{2k} Pr(\{X = m\}.Y) = \sum_{m=k+1}^{2k} Pr(X = m).Pr(Y/\{X = m\})$.

It is easy to see that $Pr(X = m) = \binom{D}{m} / \binom{N}{m}$ where $k+1 \leq m \leq 2k$.

Moreover, $Pr(Y/\{X = m\})$ is the probability to get $k+1$ identical shares in the set of size m which belongs to the set of $2k+1$ elements containing $k+1$ ones of value v and k ones of value $-v$, that is, $Pr(Y/\{X = m\}) = \binom{k+1}{k+1}\binom{k}{m-(k+1)} / \binom{2k+1}{m}$.

Consequently, $Pr(XY) = \sum_{m=k+1}^{2k}\left[\binom{D}{m}/\binom{N}{m}\right].\left[\binom{k+1}{k+1}\binom{k}{m-(k+1)}/\binom{2k+1}{m}\right] = \sum_{m=k+1}^{2k}\frac{D(D-1)...(D-m+1)}{N(N-1)...(N-m+1)}.\frac{k(k-1)...(m-k)}{(2k+1)2k...(m+1)} < \sum_{m=k+1}^{2k}\left(\frac{D}{N}\right)^m.\left(\frac{1}{2}\right)^{2k+1-m}$

Choose $\xi(k) = \sum_{m=k+1}^{2k}\left(\frac{D}{N}\right)^m.\left(\frac{1}{2}\right)^{2k+1-m}$. We can see that, if $D \ll N$, then $\xi(k)$ is negligible function for all value of k. If k is a large number, then $\xi(k)$ is also negligible. From definition 1 our protocol is private.

Corollary 1. *In the first phase, the maximum number of votes revealed by a coalition of D dishonest nodes is $2D$ with $D \leq N/5$.*

Proof. In the sharing phase, each node receives exactly $2k+1$ shares, and so, the dishonest nodes collect $D(2k+1)$ shares.

From Lemma 3, we have $D(2k+1) = (k+1)x + y$, where x is the number of honest node to be revealed and y is the size of the set of shares which does not contain any $k+1$ identical shares of any node.

Since $y \geq 0$, we have $D(2k+1) \geq x(k+1)$. It implies $x \leq \lfloor D.\frac{2k+1}{k+1} \rfloor \leq 2D$ (since $\frac{2k+1}{k+1} < 2$ for all $k > 0$).

To make sure that not all half the honest nodes are revealed, i.e., $x < N/2$, the condition of coalition $\mathcal{D}$ has to be satisfied the following condition: $D(2k+1) < \frac{N-D}{2}.(k+1)$, it means $D < \frac{N(k+1)}{5k+3}$. Because $\frac{N}{5} < \frac{N(k+1)}{5k+3} \leq \frac{N}{4}$ for all $k \geq 1$, to make sure the inequation $D < \frac{N(k+1)}{5k+3}$ always occurs, we must have the condition $D \leq N/5$.

So $x < \min\{2D, \frac{N-D}{2}\} = 2D$ with $D \leq N/5$.

Accuracy. We present the accuracy based on the ability of the honest nodes to get correct output and to control the impact from dishonest nodes. We first justify clearly the condition for receiving broadcasting messages from neighbors of each node that we use in the broadcasting phase.

As mentioned in Algorithm 1, despite the use of richer social graph structures, one node can receive/send so many duplicated messages from/to other nodes. This leads to flooding the local storage. We call this situation a naive approach. We propose a simple optimal technique like this: each node n should receive broadcasting messages from the path with length $l_s \in [\delta(s,n); \delta_L(s,n)]$. We use breadth-first expansion from s with the assumption that transmission in one edge takes one time unit. This gives the fact that node n receives the broadcasting messages from the shortest paths first, and later it receives messages from the paths which length are not greater than $\delta_L(s,n)$. Moreover, for each message generated by s and passed from the path p which length satisfies the condition $\delta(s,n) \leq l(p) \leq \delta_L(s,n)$, n checks whether the collected data c_s exists in $\mathcal{C}_n[s]$ or not. If $\mathcal{C}_n[s]$ does not contain c_s, n inserts c_s into $\mathcal{C}_n[s]$ and then forwards to other neighbors except the node it just received message. Otherwise, n does not put it into $\mathcal{C}_n[s]$ or forward it. This way of creation gives us $\mathcal{C}_n[s]$ containing only distinct values.

To show the correctness of our optimal technique, we have to prove that all of (distinct) information that n receives by using naive approach are stored in $\mathcal{C}_n[s]$ of the optimal approach and vice versa. More formally, let us denote the set of all paths between s and n, the set of possible collected data of the naive approach and of the optimal solution by $N_{s,n}$, V_1 and $O_{s,n}$, V_2, respectively. We also define the mapping $f_{s,n} : 2^{Pa(s,n)} \to \mathcal{C}_n[s]$. We have $N_{s,n} \mapsto V_1$ and $O_{s,n} \mapsto V_2$. We confirm the correctness of the optimal approach by showing that $V_1 = V_2$ in Lemma 5.

Lemma 5. *Prove that $V_1 = V_2$*

Proof. Let $c_n^s(p)$ be the collected data receiving from path $p = p(s,n)$.

We have $V_1 = \{c_n^s(p) | p \in N_{s,n}\}$, and $V_2 = \{c_n^s(p) | p \in O_{s,n}\}$

It is easy to see that $V_2 \subseteq V_1$ since $O_{s,n} \subseteq N_{s,n}$.

We now prove that $V_1 \subseteq V_2$, i.e., $\forall c \in V_1 \Rightarrow c \in V_2$, by contradiction.

Indeed, assume the contrary, i.e., $\exists \omega \in N_{s,n} : c = c_n^s(\omega) \in V_1$ and $c \notin V_2$.

Since $O_{s,n} = \{p \in Pa(s,n) | \delta(s,n) \leq l(p) \leq \delta_L(s,n)\ \}$, it infers that (i) $l(p) < \delta(s,n)$ or (ii) $l(p) > \delta_L(s,n)$.

Case (i) cannot occur since $\delta(s,n) = l(p_S(s,n)) \leq l(p), \forall p \in Pa(s,n)$. Thus, we only consider case(ii). W.l.o.g., assume $l(\omega) = l_0 + 1$ where $l_0 = \delta_L(s,n)$.

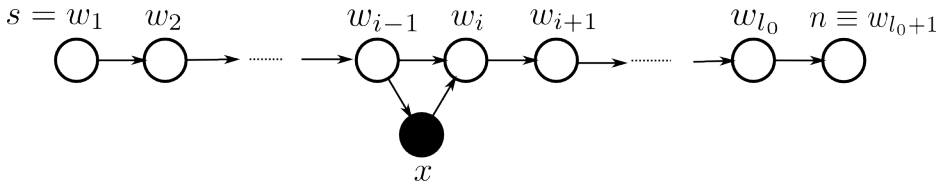


Figure 3: Path $\omega = \langle s \equiv w_1, w_2, ..., w_{i-1}, x, w_i, ..., w_{l_0+1} \equiv n \rangle$ of length $\delta_L(s,n) + 1$

Let us consider a path $p_0(s,n) = \langle s \equiv w_1, w_2, ..., w_{l_0}, w_{l_0+1} \equiv n \rangle$ (where $w_i \neq w_j$ if $i \neq j$) of length l_0 and depicted in Figure 3.

Because n gets value c but not put it into V_2, this value must be transferred by a certain node w_i in path p_0 and is also emitted from a certain node x which is not in path p_0 but connects to w_i. W.l.o.g., suppose x connects to w_{i-1} and the path ω is: $\omega = \langle s \equiv w_1, w_2, ..., w_{i-1}, x, w_i, ..., w_{l_0+1} \equiv n \rangle$. (We assume this for easily understanding the proof. In general case that x does not connect to other nodes in p_0 (except w_i) but it is contained in a certain path such as $\langle s \equiv w'_1, w'_2, ..., w'_{i-1} \rangle$, we can consider the following path: $\omega' = \langle s \equiv w'_1, w'_2, ..., w'_{i-1}, x, w_i, w_{i+1}, ..., w_{l_0+1} \equiv n \rangle$. We see that the role of ω and ω' in the proof are equivalent).

Firstly, we prove that x cannot connect to other nodes in p_0, except w_i and w_{i-1}, i.e., $e(x, w_j) = 0$, $\forall j \notin \{i, i-1\}$. Indeed, if $\exists j$: $e(x, w_j) = 1$, $i < j \leq l_0 + 1$, then existing a message from the path $p' = \langle s \equiv w_1, w_2, ..., w_{i-1}, x, w_j, w_{j+1}, ..., w_{l_0+1} \equiv n \rangle$ of length $l(p') = i - 1 + (l_0 + 1 - (j-1)) \leq l_0$ contains value c. Moreover, as $p' \in O_{s,n}$, we have $c \in V_2$. Contradiction.

So, path $\langle s, w_2, ..., w_{i-1}, x \rangle \in Pa_S(s,x)$ and path $\langle x, w_i, w_{i+1}, ..., w_{l_0+1} \equiv n \rangle \in Pa_S(x,u)$. We have: $l_0 = \delta_L(s,n) = \max_y \{\delta(s,y) + \delta(y,n)\} \geq \delta(s,x) + \delta(x,n) = i - 1 + (l_0 + 1 - (i-1)) = l_0 + 1$. This inequation gives us the contradiction. Consequently, $V_1 \subseteq V_2$.

In conclusion, $V_1 = V_2$.

Corollary 2. *If $|V_2| = 1$ then the single element is correct collected data. Otherwise, there exists an inconsistency.*

Proof. As G is honest graph, then $\exists p \in Pa(s,n)$: $T(p) > 0$, for all $s, n \in \mathcal{H}(G)$. By Lemma 5, V_2 contains at least one correct collected data, $|V_2| \geq 1$. Moreover, all values in V_2 are distinct, and there always exists one correct value, we get the conclusion of proof.

Consider the network with privacy conscious settings. In such network, each node has no knowledge to calculate bounds for the path lengths between it and other nodes. In that case, in actions 2 of the broadcasting phase, after receiving message from source s, node n could not check the length of the path delivering that message, i.e., n does not check the condition $l_s \in \big[\delta(s,n), \delta_L(s,n)\big]$. It just verifies the set $\mathcal{C}_n[s]$ and stores (and forwards, resp.) information which has never been received (and forwarded, resp.) earlier. We call this approach "privacy-conscious one". Notice that, this approach is different from the naive one. In the naive

approach, node n floods network by receiving/forwarding messages: it gets all messages, then puts c_s into $\mathcal{C}_n[s]$, and transfers it to other friends without checking whether that information is already existed in $\mathcal{C}_n[s]$.

Corollary 3. *By using privacy-conscious approach, no information from s to n is lost.*

Proof. Let V_3 be the set of possible collected data in the privacy-conscious approach. We first prove that $V_3 = V_2 = V_1$. Indeed, from the definition of V_1 for naive approach above, it is easy to see that $V_3 \subset V_1$. Moreover, in the privacy-conscious approach, node n receives all messages without checking the length of the path delivering them, and thus $V_2 \subset V_3$. By Lemma 5, $V_1 = V_2$, hence, we have $V_3 = V_2 = V_1$.

Lemma 6 (Sharing)**.** *In the sharing phase, one dishonest node affects at most $2k+2$ to the final result.*

Proof. We will prove that $\Delta(g(\mu_n), v_n) \leq 2k + 2$, where $n \in \mathcal{D}(G)$.

We have $\mu_n \subset \mathcal{P}_n$, $|\mu_n| \leq 2k + 1$. A dishonest node tries to impact the final result by either (i) generating and sending less than $2k+1$ shares, or (ii) sending less than k shares of "1" or more than $k+1$ shares of "-1". From the standpoint of dishonest coalition, the best case is when it votes $v_n = +1$ but sends all $2k + 1$ shares of value "-1", it means $\mu_n = \mathcal{P}_n = \{-1, -1, \ldots, -1\}, |\mu_n| = 2k + 1$. The impact is $\Delta(g(\mu_n), v_n) \leq |1 - (-2k - 1)| = 2k + 2$.

Lemma 7 (Computing collected data)**.** *The maximum impact of the dishonest node into the collected data is $4k + 2$.*

Proof. We will prove $\Delta(c_n, c'_n) \leq 4k + 2$ where c_n and c'_n are respectively the correct collected data and the sending value of node $n \in \mathcal{D}(G)$. Indeed, each node can only receive at most $2k+1$ shares generating from its $2k + 1$ direct neighbors. After receiving the set of shares, a dishonest node can modify that data by inverting all shares of "1" to shares of "-1" in order to decrease it as much as possible. It can also create a forged collected data. But in any case, the dishonest node can only modify and create at most $2k + 1$ shares, and $c_n \in [-2k - 1, 2k + 1]$. The best case, from the point of view of dishonest coalition, is when it gets $c_n = 2k+1$ (i.e., $2k+1$ shares of "1"), and forwards the different value $c'_n = -2k - 1$ by converting all share "1" into "-1" . Thus we have $\Delta(c_n, c'_n) \leq |2k + 1 - (-2k - 1)| = 4k + 2$.

Corollary 4. *There exists a public verification scheme detecting with certainty the attack that a dishonest node modifies the collected data of other one by more than $4k + 2$.*

Proof. As $c_n \in [-2k - 1, 2k + 1]$, if a dishonest sends an incorrect value $c'_n \notin [-2k - 1, 2k + 1]$ then $\Delta(c_n, c'_n) > 4k + 2$. The attack is identified and the dishonest node is exposed.

Lemma 8 (Broadcasting collected data)**.** *There exists a public verification scheme that detects with certainty a dishonest node broadcasts (or forwards) inconsistent copies of its collected data (or collected data of other nodes) and exposes that dishonest one.*

Proof. Before proving this Lemma, we present all capabilities of dishonest nodes in the broadcasting phase. Dishonest nodes can promote their votes by modifying the content of the broadcast/forwarded messages and try to convince that are correct. However, as motivated in Section 2.1, we do not take into account the Sybil attacks, spam, and wrongfully blaming since these kinds of attacks are already detected by several practical systems. Therefore, without Sybil attacks, in the broadcasting phase, dishonest nodes can change all contents of the broadcasting message, except identity. It implies they cannot create any forged messages which contain identity of other nodes. Moreover, in the wrongfully blaming, it may be one dishonest node or

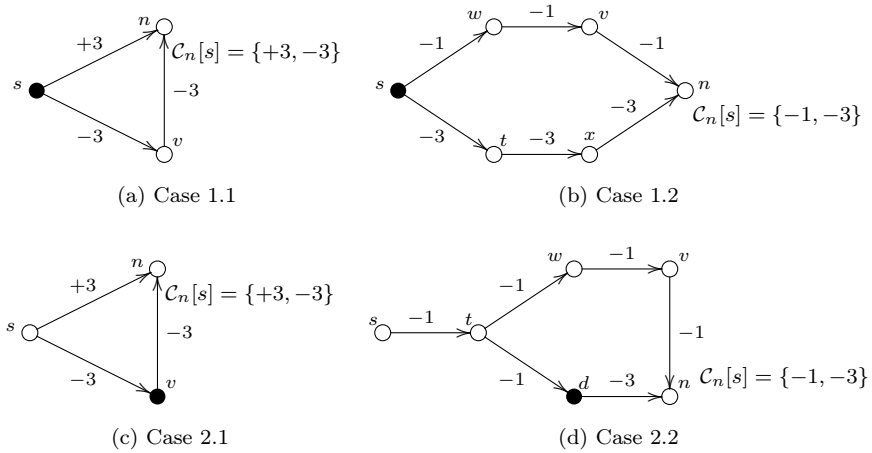
coalition try to accuse other honest node or set of other dishonest nodes. We can understand about this kind of accuse by considering an example in Figure 4a. In this case, node n receives two different values from source s, one directly from s (+3) and one from v (-3). If n requests s the value it sent to other nodes, and s replies that s sent to v value +3 (same as the one sent to n), in this case, s will indirectly wrongfully accuse v as dishonest node, because according to information from s, v later forwards to n different value from the one it received from s, and thus, v must be dishonest and tagged. Actually, in a system that honest nodes are majority, the probability for one dishonest nodes to be exposed when wrongly accusing honest ones is high. Like in the example, if v is wrongly accused only by a small number of nodes, the allegation would be in doubt and not be considered, and the accuser s would be finally backfired. By the way, we do not allow this kind of blame in the system, and assume that no node would like to be tagged as dishonest which does not wrongly blame other nodes.

Using above assumptions and all possible capabilities of dishonest nodes, we can detect the attacks in broadcasting/forwarding inconsistent copies of collected data by using the following verification scheme: requesting and checking routing tables of neighbors. More particularly, when honest node n detects inconsistency of the collected data from source s, i.e., $\mathcal{C}_n[s] = \{w_1, w_2, \ldots, w_l\}$ where $l > 1, w_i \neq w_j, i \neq j$, for each $w_i \in \mathcal{C}_n[s]$, n checks in the routing table $\Gamma_n[s]$ to finds out all neighbors $\{v_{i_1}, v_{i_2}, \ldots, v_{i_j}\}$ that send w_i to n. Then n sends $\Gamma_n[s]$ as well as requests to check the inconsistency of the value w_i to its neighbors, neighbors of neighbors, and so on. Notice that, as motivated above, we do not allow the dishonest nodes wrongfully blame honest nodes. Hence, the dishonest nodes cannot spoof incorrect versions of routing tables because that action has indirectly wrongfully blamed the honest nodes that the honest nodes have broadcast/forward wrong value they have given. With that notice, and based on the information from direct and indirect neighbors, there exists a honest node, direct or indirect neighbor of n, who can detect the violation and expose the dishonest nodes with the following cases. Here we present all of the dishonest nodes' intention, and in each case we give the specific way demonstrating in the concrete example in Figure 4, to expose correctly the dishonest one.

1. *Broadcast inconsistent copies of its collected data to honest nodes.*
   1.1. *The dishonest node forwards two different values to honest nodes which are directly connected.*
   Figure 4a shows that s forwards different value to n and v. As v is neighbor of n, it receives from v the different value emitted by s. At the time of detecting of inconsistency, n cannot conclude which node (or its neighbor) is dishonest. n requests s and v to broadcast their routing tables and it also broadcasts its own table. By doing this, n can verify that s sent different copies of its data.
   1.2. *The dishonest node forwards two different values to honest nodes but they have some common (indirect or direct) honest friends.*
   In Figure 4b, s sends two different copies to w and t. As G is a honest graph, there exists a honest node n that receives two different value and detects the inconsistency by reviewing the size of $\mathcal{C}_n[s]$. n then requests its friends v and x to broadcast their routing tables and they also request from theirs neighbors to broadcast their routing tables, and so on. By doing this, w and t will receive the routing table of each other and detect that s sent inconsistent copies to them.
2. *Forward inconsistent copies of the receiving collected data to honest nodes.*
   2.1. *The dishonest node forwards a different value from the one it received to honest node which is directly connected to the source.*
   This scenario is demonstrated in Figure 4c. Node v gets from s value 3 but it forwards value -3 to n. As n is a neighbor of s, it detects the inconsistency. Node v is exposed when each node broadcast the routing table containing data it received to each other.

2.2. *The dishonest node forwards a different value it received from one honest node to other honest node which have a honest path between them.*
This more general case, comparing to case 2.1, is depicted in Figure 4d. After receiving correct collected data from t, d forwards a forged one to n. From the honest path $\langle s, t, w, v, n\rangle$, n receives other value emitted from s. Because of inconsistency, n again requests the broadcasting of routing table to its direct and indirect neighbors. When checking its routing table, honest node t will detect the misbehavior of d: d forwarded a different value from the one t sent to.

3. *Not broadcast the collected data*: This action can also be detected easily when honest node checks and recognizes that it has not got any collected data of the certain source.



(a) Case 1.1 (b) Case 1.2

(c) Case 2.1 (d) Case 2.2

Figure 4: Intention of the dishonest nodes.
Figure 4a and 4b depict the simple and complex case that dishonest node broadcasts inconsistent copies of its collected data to honest nodes.
Figure 4c and 4d show the simple and complex case that dishonest node forwards different value from the one it received to honest nodes.

Lemma 9. *Every dishonest node may affect the expected final result up to* $6k + 4$ *without being detected by verification.*

Proof. From Lemmas 6-8, we have the impact of the dishonest node n on the final outcome of the protocol is $\Delta(\mathsf{result}_n, \sum_{i=0}^{N-1} v_i) = \Delta(\mu_n, v_n) + \Delta(c_n, c'_n) \leq (2k + 2) + (4k + 2) = 6k + 4$. Thus, $\Lambda = \frac{6k+4}{2N} = \frac{3k+2}{N}$. Choose a function $\xi(k) = \frac{3k+2}{N}$. When $k \ll N$, $\xi(k)$ is negligible. By Definition 2 we say that the protocol is accurate.

Corollary 5. *Let* α*, in average, be the proportion of nodes voting "+1". The maximum number of dishonest nodes system can tolerate so that the final result is not affected is* $\lfloor \frac{N|2\alpha-1|}{6k+4} \rfloor$.

Proof. We have $Pr[v = 1] = \alpha$, $Pr[v = -1] = 1-\alpha$ and the final output is $\alpha N.1-(1-\alpha)N.(-1) = (2\alpha-1)N$. It can be inferred from Lemma 9 that the maximum impact from dishonest coalition is $(6k + 4)D$. As mentioned in the polling model, we assume that the system wins with vote "+1", i.e., $\alpha > 0.5$, and the dishonest coalition tries to do decrease the output as much as possible. Thus, to make sure that the dishonest nodes do not affect to the final result, it implies $(6k+4)D \leq (2\alpha-1)N$ or $D \leq \lfloor \frac{(2\alpha-1)N}{6k+4} \rfloor$. On other hand side, if the system wins with vote "-1"

and dishonest coalition tries to increase the output as much as possible, we easily get the same result $D \leq \lfloor \frac{(1-2\alpha)N}{6k+4} \rfloor$ where $\alpha \leq 0.5$. Generally, we have the bound $D \leq \lfloor \frac{|2\alpha-1|N}{6k+4} \rfloor$.

Proposition 4 (Spatial complexity)**.** *The total space each node must hold is $O(N.D.d_n)$ if there exists D dishonest nodes.*

Proof. Each node u_n needs to maintain a list of d_n direct neighbors to contact, a set $\mathcal{C}_n$ to get possible collected data of other nodes, a routing table Γ_n, a set h_n size of $N-1$ to store the final choosing collected data. For $\mathcal{C}_n$, in the worst case, for each honest source u_s, dishonest coalition can forward to u_n at most D different values, and u_n also receives one correct value from the honest path. Hence, the maximum size of $\mathcal{C}_n$ is $(N-1).(D+1)$. Likewise, the worst case of $\Gamma_n[s]$ is when all shortest paths between source u_s and u_n are passed by its friends and all dishonest nodes. Since there are d_n neighbors and D honest nodes, we have the size of $\Gamma_n[s]$ is at most $D.d_n$, and thus the maximum size of Γ_n is $(N-1).D.d_n$. Consequently, the total space each node must hold in the worst case is $O(d_n)+O((N-1).(D+1))+O((N-1).D.d_n) = O(N.D.d_n)$.

Proposition 5 (Message complexity)**.** *The number of messages in our protocol is $O(N.D.d_n+k)$ if there exists D dishonest nodes.*

Proof. Node u_n sends $2k+1$ messages to its direct neighbors in the first phase, broadcasts d_n messages in the broadcasting phase to all of its direct neighbors, and hence, totally $|\mathcal{C}_n|.(d_n-1)$ messages. The worst case for $\mathcal{C}_n$ is the same as the one considered in the spatial complexity part, i.e., $|\mathcal{C}_n| = (N-1)(D+1)$, and thus, we have the total spatial complexity is $O(2k+1)+O((N-1).(D+1).(d_n-1)) = O(N.D.d_n+k)$.

Proposition 6 (Time complexity)**.** *Assuming the system is synchronous one in which time generates in round. Then the protocol operates in $O(k+N^2)$ rounds.*

Proof. The sharing phase operates in $2k+1$ rounds. In the broadcasting phase, node u_n sends collected data c_n to d_n neighbors which takes it d_n rounds. The collected data is delivered to some node u_s through the path with length at most $\delta_L(u_n, u_s)$. For each intermediate node u_t of this path, it forwards to its d_t-1 friends. In the system, all nodes send message in parallel, and this implies the collected data c_n requires $\max_{0\leq s,n<N}\{\delta_L(u_n,u_s).\max_{u_t\in p_S(u_n,u_s)}\{d_t-1\}\}$ rounds. Therefore, the time complexity is $O(k+\max_{0\leq s,n<N}\{\delta_L(u_n,u_s).\max_{u_t\in p_S(u_n,u_s)}\{d_t-1\}\} = O(k+(N-1).(N-2)) = O(k+N^2)$.

Theorem 2. *The properties of $\mathcal{G}_2$ are the necessary and sufficient condition for the polling protocol to be deployed correctly in the system containing dishonest node.*

Proof. By Lemmas 3-9, we already showed the sufficient conditions, i.e., our protocol works successfully with graphs of $\mathcal{G}_2$ and preserves its properties. We only need to clarify the remaining proof of this Theorem. Assume we have a general graph G. We approach the proof by sketching step by step the requirements G should obtain so that all properties of protocol are guaranteed.

In the sharing phase, node u_n sends (or receives) exactly $2k+1$ messages, i.e., $|\mathcal{F}_n| = |\mathcal{Q}_n| = 2k+1$. It can send (or receive, resp.) messages to (or from, resp.) the same or different neighbors. Thus, $d_n \geq 2k+1$. Notice that if these two sets are disjoint, i.e., $\mathcal{F}_n \cap \mathcal{Q}_n = \varnothing$, then u_n has at least $|\mathcal{F}_n| + |\mathcal{Q}_n| = 2(2k+1)$ neighbors. The set of the graphs with condition $d_n \geq 2k+1$ also includes the set of graphs with condition $d_n \geq 2(2k+1)$. Therefore, to apply protocol correctly, G must have the property P_{g_1}.

In addition, in the broadcasting phase of protocol, it can be inferred from Lemma 2 that, in order to make decision about the collected data of source s, node n has to check $|\mathcal{C}_n[s]|$. If $|\mathcal{C}_n[s]| > 1$ then there exists an inconsistency, and n starts the verification process. In that

procedure, it has to request its neighbors to send the routing table $\Gamma[s]$, and its neighbors do the same activities with others. The security in this situation is the protection of the table transmission so that the the honest node has some witnesses to expose the dishonest ones. As motivated in the introduction, we do not take into account the case that dishonest nodes blame wrongfully other honest ones in the sense that they cannot modify the value that honest nodes sent to or forwarded in the content of the Γ. Suppose G is not a honest graph, i.e., $\exists u, v \in \mathcal{H}(G), T(p(u,v)) = 0$. W.l.o.g, consider path $p = \langle u \equiv u_{k_1}, u_{k_2}, \ldots, u_{k_m} \equiv v \rangle$ (where $e(u_{k_i}, u_{k_{i+1}}) = 1$, $1 \leq i \leq m-1$), then $T(p) = \prod_{i=1}^{m-1} q(u_{k_i}, u_{k_{i+1}}) = 0$. It implies that $\forall p(u,v)$, $\exists u_{k_j}, u_{k_{j+1}}$: $q(u_{k_j}, u_{k_{j+1}}) = 0$, i.e., node u_{k_j} or $u_{k_{j+1}}$ is dishonest. And thus, the routing message always passes throughout a dishonest node, the security property of protocol is not preserved. Consequently, G must be a honest graph, or satisfies P_{g_2}.

Naively, assume $D \geq N/2$, if all dishonest nodes behave properly towards the system by voting a poll "-1", not creating the set of shares to forward as well as not transmitting a forged vote of other one, the final result is $\sum_n v_n < 0$ since $D \geq N/2$ regardless of other votes from honest nodes. So we know the result before running protocol. Contradiction. Therefore, G has property P_{g_3}.

In conclusion, G is the member of family of graph $\mathcal{G}_2$.

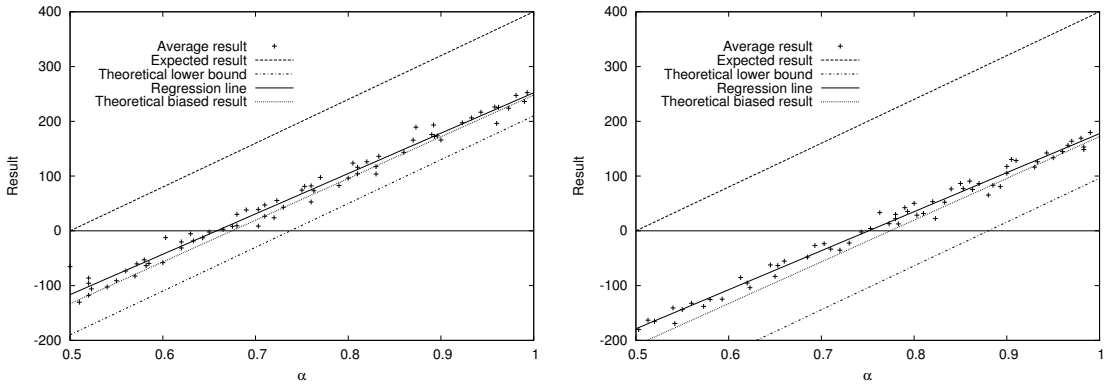
6 Experimental evaluation

We do some experiments to analyse the correctness of the protocol by observing the difference between experiment output and the theoretical one. In the experiments, we use UDP and asynchrony in message exchanging without crash or message loss. We implement protocol by using an open-source Java library, YALPS[6], to demonstrate the communication amongst node and facilitate the development and testing of the applications. The application based on this framework can be run both in simulation and real-world mode. As we do not consider crash and message loss in the experiments, we performed them in the local machine by a simulating graph. We plan later to examine protocol in the real-world network like PlanetLab, with the presence of crash and message loss to fully evaluated.

We study the experiment with the existence of dishonest nodes, and examine whether the real impact from the dishonest coalition to the outcome is inside the analytical bound or not. Moreover, we consider the worst case for the system in the sense that the dishonest nodes try to misbehave protocol by expressing the attack to decrease the final result at most without being detected: each dishonest node always sends $2k+1$ shares of value "-1" in the sharing phase, and converts all receiving shares of "1" into ones of "-1". Thus, each dishonest node affects the final result at most $2k+2$ in the sharing phase (if its desired vote is "1" and it sends $2k+1$ shares of "-1") and $2(2k+1) = 4k+2$ in the broadcasting phase (if $2k+1$ receiving shares are "1" and all are converted to "-1"). In other words, total impact will be up to $6k+4$. If we denote by α number of nodes voting "+1", then the expected result will be $\alpha N - (1-\alpha)N = (2\alpha - 1)N$. So theoretically, the biased final outcome should be inside the interval $[(2\alpha-1)N - (6k+4)D; (2\alpha-1)N]$.

We examine the same experiment condition about number of nodes and dishonest nodes as [9, 10]. Without loss of generality, we consider α value in the interval $[0.5, 1.0]$. Figure 5 depicts our experiments for the network with $N = 400$, $D = 19$ in two subcases corresponding to two different values of privacy parameter $k = 1$ (in Figure 5a) and $k = 2$ (in Figure 5b). In each test, we compute the average output amongst all nodes and represented it as a point in the figures. We see that the experimental result is certainly inside two bounds, expected theoretical bound (thick-dashed line) and lower bound (dot-dashed line). The experiment result never touches the

[6] http://yalps.gforge.inria.fr/



(a) $k = 1$ (b) $k = 2$

Figure 5: Experiment with $N = 400$, $D = 19$ to check the accuracy of protocol.

expected line as the dishonest node always sends $2k + 1$ shares of "-1" and converts all receiving shares of "1" to "-1" and this decreases the final output and makes it less than the expected outcome. Moreover, the experiment result does not reach the theoretical lower bound either, i.e., the average impact is less than $6k + 4$. The reason we obtained this consequence comes from the fact that the amount of average impact from dishonest nodes depends on the number of shares "+1" it receives from neighbors. Namely, in the first phase, nodes voting "+1" will send $k + 1$ shares of "+1" and k shares of "-1", and vice versa. Therefore, the probability one node receives shares of "+1" from its neighbors is $\alpha.\frac{k+1}{2k+1} + (1 - \alpha).\frac{k}{2k+1} = \frac{k+\alpha}{2k+1}$, and the average number of shares of "+1" is $(2k + 1).\frac{k+\alpha}{2k+1} = k + \alpha$. From the position of the dishonest nodes, they convert all shares of "+1" into "-1" and thus, the impact will be $2(k + \alpha)$. Combining to the fact that they always distribute $2k + 1$ shares of "-1" with the maximum impact is $2k + 2$ (if its desired vote is "+1"), we achieve the total impact $2k + 2 + 2(k + \alpha) = 4k + 2\alpha + 2$. In conclusion, with D dishonest nodes, the average biased outcome is $(2\alpha - 1)N - (4k + 2\alpha + 2)D$. In Figure 5, we present this average value in a thin-dotted line. We try to fit our data points with a regression line $a(2\alpha - 1) - b(4k + 2\alpha + 2)$ and get these results (depicted as a solid line in Figure 5): for $k = 1$: $a = 385$ and $b = 17$, and for $k = 2$: $a = 373$ and $b = 16$. These parameters are quite accurate comparing to conditions of network with $N = 400$ and $D = 19$.

In Figure 5, we see that the impact from the dishonest nodes in case $k = 2$ is greater than in case $k = 1$. This result is reasonable since we know that the higher value k is, the higher privacy can be hold but the higher impact dishonest nodes can enforce, and hence, the worse the final outcome is. Besides, all nodes output the correct result (or the final result greater than 0) for $k = 1$ when $\alpha \geq 0.67$, and for $k = 2$ when $\alpha \geq 0.75$. It means the dishonest nodes confuse the majority of nodes for $k = 1$ when $\alpha < 0.67$, and for $k = 2$ when $\alpha < 0.75$. Comparing to other recent polling protocols like [9, 10], that value of α in our experiment is similar to them.

7 Related work

In this section, we discuss some recent research and motivate our contribution. Secret sharing scheme in which a secret is divided into many shares and distributed to mutually suspicious

nodes is first independently proposed by Blackley and Shamir. Till now, a lot of work related to this topic has been published. We here describe a few of the most influential solutions which does not depend on the heavy mathematical computations like cryptography.

DPol [9, 10]. This is a simple distributed polling protocol in a social network without using cryptography where nodes are concerned about their reputation. It ensures privacy and accuracy despite the presence of dishonest nodes by means of combination of secret sharing and verification procedures. In DPol, no user has a particular role and this views as a peer-to-peer network, and increase the scalability and robustness. This work suggested a model of adversaries to be compelling for various non-critical private and secure distributed computation problems in social settings and DPol thus introduces a new direction of research in distributed computing. By the way, DPol also remains some shortcomings when applied in practice. Firstly, DPol relies on a structured overlay, cluster-ring-based structure, which is on top and really apart from the social graph. It does not keep the normal social graph and take into account any social links between nodes in the sense that it uses the uniform assignment of nodes to group. This is not practical as we have to target a special one using notion of group instead of reserving the normal structure of the graph. Till now, we do not know the way to build such an overlay ring-structure from a social graph in a decentralized way. Moreover, the number of nodes should be a perfect square such that a graph G_P with N nodes can be divided into $\sqrt{N}$ groups of size $\sqrt{N}$. On the contrary, we propose a protocol deployed in a more general structure. Our objective is to keep the natural property of the graph in the sense that the node and social links of graph should be preserved, and each individual can perform the voting process privately and secure by its self, not based on the group division. We can show that the family of graph $\mathcal{G}_2$ (we do not mention $\mathcal{G}_1$ here as we just take care the general case of graph which has dishonest and honest nodes) considering in our work includes the particular graph G_P applied in DPol, i.e., $G_P \subseteq \mathcal{G}_2$, with the following reasons:

1. In DPol, each node u_n maintains these sets of friends: a set P_o of its officemates inside the same group g, a fixed-size set $\mathcal{F}_n$ of proxies for sending messages, and a fixed-size set $\mathcal{Q}_n$ of clients for receiving messages where $|\mathcal{F}_n| = |\mathcal{Q}_n| = 2k + 1$. Of course the degree of each node is not less than $2k + 1$. This requirement is satisfied property P_{g_1} of our graph $\mathcal{G}_2$.

2. DPol also gives the fact that the dishonest coalition can both perform and cover dishonest actions as soon as the number of dishonest nodes in two consecutive groups is not less than $\sqrt{N}$. Thus, it needs the requirement $|g_m \cap \mathcal{D}| + |g_{m+1} \cap \mathcal{D}| < \sqrt{N}$ holds for all group g_m. If $D < \sqrt{N}$, this condition holds with certainty. If $D \geq N/2$, the coalition compromises the system with certainty. And if $\sqrt{N} \leq D < N/2$, the probability to do that intention is less than 1. In our case, we use the same condition $D < N/2$ to make sure that there is a majority of honest nodes in the graph and protocol works properly. The condition $D \leq N/5$ is to guarantee the number of revealed votes is not greater than half of honest nodes and this one includes the condition $D < \sqrt{N}$ which DPol used, and thus, DPol satisfies the property P_{g_3}.

3. The property P_{g_2} of our graph $\mathcal{G}_2$ is implicitly given in G_P, i.e., $\forall u, v \in \mathcal{H}(G_P) : T(u, v) > 0$. Indeed, the condition $D < \sqrt{N}$ gives the fact that the number of dishonest nodes is strictly smaller than the size of a group. This ensures the existence of honest node in a group even the worst case that all dishonest nodes are concentrated in one or two consecutive groups. Suppose $u \in \mathcal{H}(g_i)$, $v \in \mathcal{H}(g_j)$. If $i = j$ then $T(u, v) = e(u, v) = 1$ since g_i is a clique. If $i \neq j$, w.l.o.g., assume $i < j$, $j = i + l$. We depict one example in Figure 6. For any two consecutive groups g_m, g_{m+1}, $\exists v_1 \in \mathcal{H}(g_m), v_2 \in \mathcal{H}(g_{m+1})$ such that $e(v_1, v_2) = 1$. This result is inferred from the fact that $|g_m \cap \mathcal{D}| + |g_{m+1} \cap \mathcal{D}| < \sqrt{N}$. Thus, $\forall x \in \mathcal{H}(g_m)$,

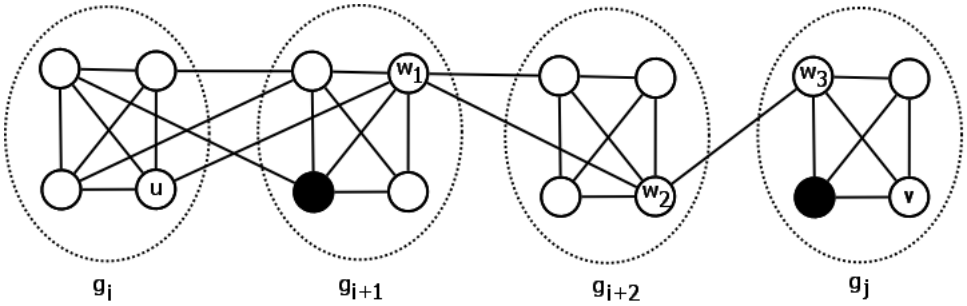


Figure 6: Existence of the honest paths between two arbitrary honest nodes

$y \in \mathcal{H}(g_{m+1})$, we have: (i) $T(x,y) = q(x,y) = 1$ if $x = v_1$, $y = v_2$; or (ii) $T(x,y) = q(x,v_1).q(v_1,y) = 1$ if $x \neq v_1$, $y = v_2$; or (iii) $T(x,y) = q(x,v_1).q(v_1,v_2).q(v_2,y) = 1$ if $x \neq v_1, y \neq v_2$. It means $T(x,y) = 1$, for all $x \in g_m$, $y \in g_{m+1}$. Apply this for a chain of two consecutive groups g_i and g_{i+1}, g_{i+1} and g_{i+2}, ..., g_{j-1} and g_j, we have $T(u,v) = T(u,w_1).T(w_1,w_2)\dots T(w_l,u) = 1$ where $w_t \in \mathcal{H}(g_{i+t})$, $1 \leq t \leq l$. Consequently, G_P is a honest graph.

Above we already convinced that $G_P \subseteq \mathcal{G}_2$, i.e., our protocol can be used in the ring-based structure G_P. We now point out that there exists a graph G_0 such that our protocol can be utilized but DPol cannot. Formally, we prove that $\exists G_0 \colon G_0 \in \mathcal{G}_2 \wedge G_0 \notin G_P$, and from this we conclude $\mathcal{G}_2 \supset G_P$. In other words, our protocol can be deployed in the graph structure which is more general than ring-based structure demonstrated in DPol. Namely, we see that, in G_P, the set of neighbors for sending message $\mathcal{F}_n$ and the set of neighbors for receiving message $\mathcal{Q}_n$ of each node are separate. In our work, $\mathcal{G}_2$ does not have that condition in the sense that each node can send and receive message on the same or different link. For instance, let us consider the graph G_0 satisfying properties P_{g_2}, P_{g_3} of $\mathcal{G}_2$ and two other conditions: $2k+1 < N \leq 2(2k+1)$ and the set of receivers $\mathcal{F}_n$ of node u_n is the output of the function:

$$\begin{aligned} f\colon A &\to 2^A \\ n &\mapsto \{(n+1) \bmod N, (n+2) \bmod N, \dots, (n+2k+1) \bmod N\} \end{aligned}$$

where $A = \{0,1,2\dots,(N-1)\}$. Figure 7 presents G_0 with $N = 6$ and $k = 1$, and the arrow expresses the direction of sending message. We will prove that $G_0 \in \mathcal{G}_2$.

Clearly, $|\mathcal{F}_n| = |f(n)| = 2k+1$. In addition, we easily see that $\mathcal{Q}_n = \{(n-1) \bmod N, (n-2) \bmod N, \dots, (n-2k-1) \bmod N\}$ and $|\mathcal{Q}_n| = 2k+1$. This follows that $d_n \geq 2k+1$ and so, G_0 also has property P_{g_1}. It infers $G_0 \in \mathcal{G}_2$.

In addition, we will clarify that $G_0 \notin G_P$ by justifying that in G_0 we have $\mathcal{F}_n \cap \mathcal{Q}_n \neq \varnothing$, i.e., $\exists i \neq n$ such that $i \in f(n)$ and $n \in f(i)$. Indeed, for $0 \leq n \leq N-1$, we have: $f(n) = \{(n+1) \bmod N, (n+2) \bmod N, \dots, (n+2k+1) \bmod N\}$. For all $1 \leq x \leq 2k+1 < N$, then $(n+x) \bmod N \neq n$. It infers that $\forall i \in f(n)$, $i \neq n$ is always true.

We just show that existing one value $i \in f(n)$ such that $f(i) \ni n$. Choose $i = [n+N-(2k+1)] \bmod N$. From the assumption that $2k+1 < N \leq 2(2k+1)$, then $n+1 \leq n+N-(2k+1) \leq n+(2k+1)$, and thus $i \in f(n)$. We have:

$$\begin{aligned} f(i) =& \big\{(i+1) \bmod N, (i+2) \bmod N, \dots, (i+2k+1) \bmod N\big\} \\ =& \big\{[n+N-(2k+1)+1] \bmod N, [n+N-(2k+1)+2] \bmod N, \dots, (n+N) \bmod N\big\} \\ =& \big\{[n+N-(2k+1)+1] \bmod N, [n+N-(2k+1)+2] \bmod N, \dots, n\big\} \end{aligned}$$

So $n \in f(i)$.

Besides, instead of showing exactly value of i, we can see that:

$$\begin{aligned}\bigcup_{i \in f(n)} f(i) =& \big\{(n+2) \bmod N, \ldots, (n+2k+2) \bmod N\big\} \cup \\ & \cup \big\{(n+3) \bmod N, \ldots, (n+2k+3) \bmod N\big\} \cup \\ & \cup \ldots \cup \big\{(n+2k+2) \bmod N, \ldots, [(n+2k+1)+(2k+1)] \bmod N\big\} \\ =& \big\{(n+2) \bmod N, (n+3) \bmod N, \ldots, (n+N) \bmod N, \ldots, [n+2(2k+1)] \bmod N\big\} \\ =& \big\{(n+2) \bmod N, (n+3) \bmod N, \ldots, n, \ldots, [n+2(2k+1)] \bmod N\big\}\end{aligned}$$

From this statement, we are sure that there exists $i \in f(n)$ such that $n \in f(i)$, and thus, $G_0 \notin G_P$ but $G_0 \in \mathcal{G}_2$. This proof shows that DPol cannot be deployed in this kind of graph, but our protocol can work.

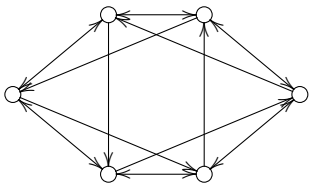


Figure 7: Direction of communication amongst nodes in our protocol

AG-S3 [7]. Giurgiu *et al.* defined the Scalable Secure Computing problem in a distributed social network called S^3 problem. Such computation problem is challenging as it relates to the situation that nodes need to compute a function $f : V \to U$ of their inputs in a set of constant size, in scalable, private and accurate way. Base on S^3 problem, they presented a distributed protocol, AG-S3, that computed a class of aggregation functions in a S^3 manner. On one hand side, this secret sharing scheme is suitable for polling protocol as its core has some strong characteristics such as: S^3 computation concerning a sequence of message exchanges and local computation such that the honest nodes eventually get the expected final result; node is allowed to obfuscate their inputs which preserves privacy; and node can do a verification process to potentially tags the dishonest nodes' profiles. On the other hand, AG-S3 uses a special overlay graph structure like DPol where N nodes are distributed into groups of size $\sqrt{N}$. This is the same drawback as DPol we already analyzed above.

Secret sharing homomorphisms [2]. This is one of the first works about secret sharing scheme. It proposed a model which described a homomorphism property to allow multiple secrets to be combined by direct computing on shares. The homomorphism property with respect to addition allows this scheme to be applied as a fault-tolerance method of holding verifiable secret-ballot elections. Nonetheless, it did not give the protection for the initial shares with the existence of dishonest nodes, and so, the final result is likely impacted.

Verifialable secret sharing scheme (VSS) and Multi-party computation protocol (MPC) [15] . VSS is an extension of [2]. This protocol works under the assumption that a majority of the nodes are honest (i.e., it can tolerate up to $D < N/2$ dishonest nodes). The secrecy achieved is unconditional and does not rely on any assumption about computational intractability. Based on VSS, they also proposed a secure MPC protocol to privately compute the user's shares and get the output with the exponentially small probability of error. However,

these techniques assume the existence of fully connected network, complete synchronous communications together with a broadcast channel, secure pairwise communication channels between participants, and particularly, heavyweight computation in mathematics. Furthermore, in spite of computing the output with negligible error comparing to the correct value, MPC does not give any condition to the user's input, so it allows dishonest node to shares an arbitrary data, even a large number. This activities can affect the output in a potentially unbounded way. Additionally, the scheme ensures the privacy of all nodes, including the dishonest ones. This prevents it from being applied into polling problem. Other later researches based on MPC such as [4, 3] have improved some aspects of this scheme. They focus on the scalability and usability properties. which the most relevant goal is minimizing the growth of complexity with number of nodes. [4] proposed the unconditionally secure protocol with the most relevant goal that is minimizing the growth of complexity. It shows that part of the communication complexity depending on the circuit size is linear in N (with $D < N/3$). Meanwhile [3] presents a general MPC protocol which is simultaneously optimal, up to lower-order terms, with respect to both efficiency and resilience. Although these studies guarantee the privacy for node, again, like [15], including the dishonest ones, and thus, this makes it hardly suitable to be used in polling.

Anonymous Multi Party Computation (AMPC) [13]. Malkhi & Palov introduced AMPC which provided users with electronic anonymity without using any conventional cryptography, or any means of non-trivial maths. Yet this method requires the assumption about existing of secure channels between any pair of honest users and under a suitable resilience threshold of dishonest users $D < \sqrt{N}$. Moreover, although this technique increases privacy, the AMPC network is built on top of normal social graph in the sense that it uses group notion to form some levels (rows): users split their inputs into several shares and submit each share to one of the nodes in the first (top) row; the top row nodes permute the input and split into several sub-shares and send each sub-share to a node in the second level; each node in the second row combines the receiving sub-shares, permutes them and continue sending the split shares to the nodes in the third level, and so on; the last (bottom) row combines the shares to obtain the output. This structure is not practical as mentioned above, and is different to our concern that we target to the normal structure of graph, not overlay or transformation of the graph.

E-voting protocol [12, 1]. Based on AMPC [13] and an extension of Rabin and Ben-Or's check vectors [15], Malkhi *et al.* proposed a distributed e-voting protocol. Despite this protocol is information-theoretically secure with strong property that system can withstand the corruption from almost half the number of nodes, each node in this protocol needs to maintain some predefined roles such as receivers, voters, dealers, talliers, registrars. This may decrease the scalability as a small set of nodes that are not part of the system (e.g., dealers) may get the load of distributing initial shares to voters, and may also decrease the robustness if specific node is failed. In addition, that protocol fully supports the requirement of democracy (guaranteeing that each voter is able to vote), verifiability, and unconditional accuracy whereas polling protocol needs to be simple by relaxing such constraints. Baudron *et al.* also proposed other distributed e-election in [1] that guarantees privacy of node, provides public verification, and robustness. However, it uses asymmetric cryptography.

Rating system. There are some similarities between reputation (rating) system and polling system in which user evaluates the quality of one person by locally promoting his choices and then collecting ideas from the remaining ones of the network. There are some works related to this concern. [16] addressed the system requirements that a reputation system should have and detailed some main attacks it can suffer. [11] designed mechanism for accurately grading and fast retrieval. However, these studies concentrated in researching frading mechanism instead of providing efficient polling schemes. Dutta *et al.* [6] discussed a distributed rating mechanism

along with rating validation schemes aimed at tackling the free-riders problem and potential collusion problem in P2P network. By the way, these approaches generally do not address to privacy, as well as not provide a global polling since grading expect on only a subset of nodes (not all nodes in the system). This makes it difficult to be applied in polling.

8 Conclusion

In this paper, we proposed a design of a distributed polling protocol and defined a family of social graphs. We proved the structures of our family of graphs constitute necessary and sufficient condition to assure privacy and accuracy properties of the protocol with the presence of dishonest nodes. To detect dishonest nodes' misbehaviors, we presented verification procedures by using routing table and shortest path scheme. Furthermore, a small but useful technique based on shortest path scheme was introduced to prevent a node from receiving/sending so many duplicated messages without losing any necessary information. Unlike other works, we considered a protocol with a more general family of graphs, but obtained some similar results. More specifically, we achieved the same maximum number of votes that dishonest coalition can reveal, and the same impact from the coalition to the final output. In the future work, we plan to design an efficient polling protocol that can be deployed in the real-world network with the presence of failure and message loss.

References

[1] O. Baudron, P.-A. Fouque, D. Pointcheval, J. Stern, and G. Poupard. Practical multi-candidate election system. In *PODC*, pages 274–283, 2001.

[2] J. C. Benaloh. Secret sharing homomorphisms: Keeping shares of a secret sharing. In A. M. Odlyzko, editor, *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 251–260. Springer, 1986.

[3] I. Damgård, Y. Ishai, M. Krøigaard, J. B. Nielsen, and A. Smith. Scalable multiparty computation with nearly optimal work and resilience. In D. Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 241–261. Springer, 2008.

[4] I. Damgård and J. B. Nielsen. Scalable and unconditionally secure multiparty computation. In A. Menezes, editor, *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 572–590. Springer, 2007.

[5] C. Delporte-Gallet, H. Fauconnier, R. Guerraoui, and E. Ruppert. Secretive birds: Privacy in population protocols. In *OPODIS*, pages 329–342, 2007.

[6] D. Dutta, A. Goel, R. Govindan, and H. Zhang. The design of a distributed rating scheme for peer-to-peer systems. 2003.

[7] A. Giurgiu, R. Guerraoui, K. Huguenin, and A.-M. Kermarrec. Computing in Social Networks. In *SSS*, pages 332–346, 2010.

[8] A. Gubichev, S. J. Bedathur, S. Seufert, and G. Weikum. Fast and accurate estimation of shortest paths in large graphs. In *CIKM*, pages 499–508, 2010.

[9] R. Guerraoui, K. Huguenin, A.-M. Kermarrec, and M. Monod. Decentralized Polling with Respectable Participants. In *OPODIS*, pages 144–158, 2009.

[10] R. Guerraoui, K. Huguenin, A.-M. Kermarrec, M. Monod, and Y. Vigfusson. Decentralized polling with respectable participants. *J. Parallel Distrib. Comput.*, 72(1):13–26, 2012.

[11] M. Gupta, P. Judge, and M. Ammar. A reputation system for peer-to-peer networks. In *Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video*, NOSSDAV '03, pages 144–152, New York, NY, USA, 2003. ACM.

[12] D. Malkhi, O. Margo, and E. Pavlov. E-voting without 'cryptography'. In M. Blaze, editor, *Financial Cryptography*, volume 2357 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2002.

[13] D. Malkhi and E. Pavlov. Anonymity without 'cryptography'. In P. F. Syverson, editor, *Financial Cryptography*, volume 2339 of *Lecture Notes in Computer Science*, pages 108–126. Springer, 2001.

[14] A. Mislove, A. Post, P. Druschel, and P. K. Gummadi. Ostra: Leveraging trust to thwart unwanted communication. In J. Crowcroft and M. Dahlin, editors, *NSDI*, pages 15–30. USENIX Association, 2008.

[15] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In D. S. Johnson, editor, *STOC*, pages 73–85. ACM, 1989.

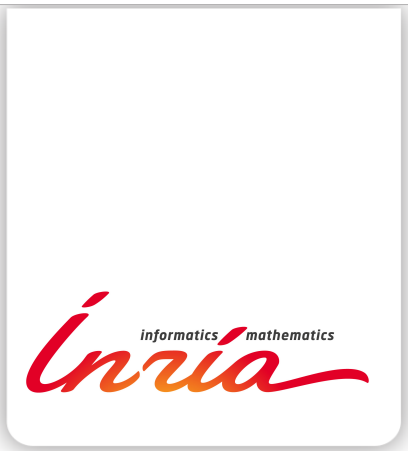
[16] M. Rodriguez-Perez, O. Esparza, and J. L. Muñoz. Analysis of peer-to-peer distributed reputation schemes. In *CollaborateCom*. IEEE, 2005.

[17] M. Sirivianos, K. Kim, and X. Yang. Socialfilter: Introducing social trust to collaborative spam mitigation. In *INFOCOM*, pages 2300–2308, 2011.

[18] D. N. Tran, B. Min, J. Li, and L. Subramanian. Sybil-resilient online content voting. In J. Rexford and E. G. Sirer, editors, *NSDI*, pages 15–28. USENIX Association, 2009.

[19] H. Yu, P. B. Gibbons, M. Kaminsky, and F. Xiao. Sybillimit: A near-optimal social network defense against sybil attacks. *IEEE/ACM Trans. Netw.*, 18(3):885–898, 2010.

[20] H. Yu, M. Kaminsky, P. B. Gibbons, and A. D. Flaxman. Sybilguard: defending against sybil attacks via social networks. *IEEE/ACM Trans. Netw.*, 16(3):576–589, 2008.



RESEARCH CENTRE
NANCY – GRAND EST

615 rue du Jardin Botanique
CS20101
54603 Villers-lès-Nancy Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399